PROCEEDINGS
OF THE 15TH INTERNATIONAL CONFERENCE "LINGUISTIC RESOURCES AND TOOLS FOR NATURAL LANGUAGE PROCESSING", ONLINE, 14-15 DECEMBER 2020

Editors

Verginica Barbu Mititelu

Elena Irimia

Dan Tufiș

Dan Cristea

Organisers

"Mihai Drăgănescu" Research Institute for Artificial Intelligence
Romanian Academy, Bucharest

Faculty of Computer Science
"Alexandru Ioan Cuza" University of Iași

Institute for Computer Science
Romanian Academy, Iași

Romanian Association of Computational Linguistics

Under the auspices of the Academy of Technical Sciences

The publication of this volume was supported by
the Faculty of Computer Science,
“Alexandru Ioan Cuza” University of Iași

ISSN 1843-911X

PROGRAMME COMMITTEE

Verginica Barbu Mititelu, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest
Tiberiu Boroș, Adobe Bucharest
Alexandru Ceaușu, European Commisssion
Mihaela Colhon, Faculty of Mathematics and Natural Sciences, University of Craiova
Svetlana Cojocaru, Institute of Mathematics and Computer Science, Academy of Sciences of Moldova, Chișinău
Horia Cucu, Faculty of Electronics, Telecommunications and Information Technology, University Politehnica of Bucharest
Dan Cristea, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iaşi and Institute for Computer Science, Romanian Academy, Iași Branch
Nils Diewald, Leibniz-Institut für Deutsche Sprache, Mannheim, Germany
Tsvetana Dimitrova, Institute for Bulgarian Language, Bulgarian Academy of Sciences
Ștefan Daniel Dumitrescu, Adobe Bucharest
Daniela Gîfu, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iaşi and Institute for Computer Science, Romanian Academy, Iași Branch
Florentina Hristea, Faculty of Mathematics and Computer Science, University of Bucharest
Adrian Iftene, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași
Diana Inkpen, School of Electrical Engineering and Computer Science, University of Ottawa, Canada
Radu Ion, Research Institute for Artificial Intelligence, Romanian Academy, Bucharest
Elena Irimia, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest
Svetlozara Leseva, Institute for Bulgarian Language, Bulgarian Academy of Sciences
Dana Lupșa, Faculty of Mathematics and Computer Science, Babeș-Balyai University
Maria Mitrofan, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest
Alex Moruz, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași and Institute for Computer Science, Romanian Academy, Iași Branch
Mihaela Onofrei, Institute for Computer Science, Romanian Academy, Iași Branch and Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași
Vasile Florian Păiș, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest
Traian Rebedea, Faculty of Automatic Control and Computers, University Politehnica of Bucharest
Adriana Stan, Faculty of Mathematics and Computer Science, "Babes-Bolyai" University of Cluj-Napoca
Elena Isabelle Tamba, "A. Philippide" Institute of Romanian Philology, Romanian Academy, Iași Branch
Horia-Nicolai Teodorescu, Institute for Computer Science, Romanian Academy, Iași Branch and "Gheorghe Asachi" Technical University of Iași
Diana Trandabăț, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași and Institute for Computer Science, Romanian Academy, Iași Branch
Ștefan Trăușan-Matu, Faculty of Automation, Control and Computer Engineering, University Politehnica of Bucharest and Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest
Dan Tufiș, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest

ORGANIZING COMMITTEE

Verginica Barbu Mititelu, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest
Dan Cristea, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași and Institute for Computer Science, Romanian Academy, Iași Branch
Lucian Gâdioi, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași
Daniela Gîfu, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași and Institute for Computer Science, Romanian Academy, Iași Branch
Adrian Iftene, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași
Elena Irimia, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest
Mihaela Onofrei, Institute for Computer Science, Romanian Academy, Iași Branch and Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași
Ionuț Pistol, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași
Andrei Scutelnicu, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași and Institute for Computer Science, Romanian Academy, Iași Branch
Diana Trandabăț, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iași
Dan Tufiș, Research Institute for Artificial Intelligence "Mihai Drăgănescu", Romanian Academy, Bucharest

TABLE OF CONTENTS

FOREWORD

This volume includes the papers of the 15th edition of ConsILR, the International Conference on Linguistic Resources and Natural Language Processing Tools, held between 14-16 December 2020, together with the 4th (and last) Workshop of the project "ReTeRom – Resources and Technologies for the Development of Human-Machine Interfaces in Language". The scientific events were organized by two institutes of the Romanian Academy, the Institute of Artificial Intelligence "Mihai Drăgănescu" in Bucharest and the Institute of Computer Science in Iași, together with the Faculty of Computer Science of the University "Alexandru Ioan Cuza" Iași and the Romanian Association of Computational Linguistics. As in most of the previous editions, the event ran under the auspices of the Technical Sciences Academy of Romania.

Organizing the ConsILR conference under the pandemic conditions was really challenging. More than once, in the past, the presentations were lively broadcast on the web during the events, but this edition, enforced by the Covid-19 pandemics, was the first when the communication was entirely online. This completely new format of the conference, to our great satisfaction, was well received by all attendants, fact proved by having one of the most numerous audiences in the whole range of ConsILR events. We are grateful to all the virtual attendants of this 15th ConsILR Conference, to the reviewers who ensured a quality selection of the papers and to the organizers who skilfully managed the online event.

From its first edition, in 2001, the ConsILR Conference (traditional acronym for *Consorțiul de Informatizare pentru Limba Română*, an initiative born in the Section for Information Science and Technology of the Romanian Academy) was meant as a meeting place for linguists and computational linguists, but also for researchers of the humanities, PhD students and master students in Computational Linguistics, all with interest in the study of the Romanian language from a computational perspective. The series of events have run, with few exceptions, every year, first in the format of a workshop, and since 2010 as an international conference. Thus, ConsILR does not strictly address researchers working on the Romanian language, but also to other scientists, from any part of the world, which could find sources of inspiration in the models and techniques developed for our language and apply them for their own languages. Opening the gate for researchers working on languages other than Romanian to participate in the Conference and publish their work in the Proceedings, a reverse influence is also facilitated, namely that their work inspires scientists working on the Romanian language.

The conference program, mirrored in this volume, was dense, with 17 presentations of the latest results of researchers in this field. The contributed articles were organized in four chapters: 1. Language Resources Development, Standardization and Exploitation, 2. Tools for Natural Language Processing, 3. Speech Recognition and Synthesis, 4. Applications. The addressed topics are of real interest, from corpora and banks of syntactic trees, models, algorithms for the most important phases of natural language processing, as well as standardized methods of representation of linguistic resources, hardware and software infrastructures for speech and textual language processing. The fourth part of the volume includes papers presenting a wide range of applications of speech recognition and synthesis for Romanian language, recognition of false news, assistance for visually impaired patients or the elderly, etc. Most of the

results presented at the conference are public, open to anyone interested in taking them over and using them.

The organisers invited five well-known researchers in our field, who accepted to deliver keynote speeches:

- Dr. Corneliu Burileanu, Professor at the Faculty of Electronics, Telecommunications and Information Technology, Vice President of University "Politehnica" of Bucharest, founder in 1984 of the Research Group in Speech Technologies, currently known as SPeeD (Speech and Dialogue Laboratory), coordinator of the biannual international conference SpeD.
- Dr. Mircea Giurgiu, Professor in the Department of Communications, Faculty of Electronics, Telecommunications and Information Technology, Technical University of Cluj-Napoca, coordinator of the Speech Processing Laboratory, with prestigious results in automatic speech synthesis.
- Dr. Daniel Marcu, a great personality in the field of language technologies. Known for his exceptional contributions, both as a scientist and as an entrepreneur, he is currently the director of Applied Sciences at Amazon, the coordinator of the teams that develop the famous ALEXA communication systems and Amazon Translate machine translation.
- Dr. Rada Mihalcea, Professor of Computer Science and Engineering at the University of Michigan and Director of the Artificial Intelligence Laboratory at the University of Michigan, Honorary Citizen of the city of Cluj-Napoca. She is the holder of numerous awards and distinctions for scientific results, being the coordinator, among others, of projects for the detection of false declarations and false news, systems that have accurately surpassed human evaluators.
- Dr. Roberto Navigli, Professor of Computer Science at Sapienza University in Rome and the leader of the University's research group in the field of Natural Language Processing. Winner of several international awards (Prof. Navigli is one of the few winners of two ERC grants), he is the coordinator of BabelNet's high-impact international projects, BabelScape.

The combination between the brightness of the spirits of this exquisite range of researchers and the quality of papers accepted for regular presentations made this event one of the most remarkable in the whole series.

December 2020
The editors

INVITED SPEAKERS

ABSTRACTS

MACHINE LEARNING – UNIVERSAL PANACEA?

CORNELIU BURILEANU

Faculty of Electronics, Telecommunications and Information Technology, University Politehnica of Bucharest

corneliu.burileanu@upb.ro

In my previous presentation at the ConsILR 2018 Conference I pointed out some of the main research directions for the "SpeeD" team. Now I am able to give more details about some achievements in several areas of interest: emotions recognition from speech, DNN approach to Romanian speech and speaker recognition, automatic music transcription, EEG classifier based on deep neural network, real-time EMG-based gesture recognition system, deep learning system for improved segmentation of lesions related to covid-19 chest CT scans, analysis of seismic waves. What do these seemingly very different areas have in common?

I am trying to demonstrate that the methods offered by machine learning could provide viable solutions for the most diverse applications. But it is also an opportunity to share some of the achievements of the team I am working with.

DEVELOPMENTS ON TEXT TO SPEECH SYNTHESIS

MIRCEA GIURGIU

Faculty of Electronics and Telecommunications, Technical University of Cluj-Napoca

mircea.giurgiu@com.utcluj.ro

The presentation will focus on an important research topic developed in the last decade at the Speech Processing Research Group from Technical University of Cluj-Napoca: text to speech synthesis for Romanian. While the earlier achievements in this field were related to speech synthesis using diphone concatenation or statistical methods, much effort has been also dedicated to speech synthesis using Deep Neural Networks (DNN). Starting from a successful end to end approach, that is training the network only with the text – audio pair, without any other text annotation, we show that there is still room for speech quality improvement from both perspectives: text processing modules, as

well as acoustic modelling. First, this has been realised through several text annotations: phonetic transcription, syllabification, lexical stress positioning, POS tagging, or even by a higher level of representations, such as text style information. The methods and the performance for these text processing modules are presented. Second, a number of investigations have been accomplished to experiment various neural network architectures for acoustic modelling in order to enhance the speech quality. For example: Tacotron 2 for expressive speech synthesis, an improved DCTTS implementation for speaker adaptation, or Tacotron for speech synthesis trained with imperfect data. Further work will conclude the presentation.

CUSTOMER OBSESSED SCIENCE

DANIEL MARCU

Amazon, daniel.marcu@gmail.com

Advancing the state of the art in the context of products and services used by hundreds of millions of customers poses challenges that go beyond those associated with advancing the state of the art in customer-free settings. In this talk, I will highlight some of these challenges and discuss approaches to overcoming them in the context of two Amazon services: Amazon Translate and Alexa.

CHALLENGES (AND OPPORTUNITIES) IN MULTIMODAL SENSING OF HUMAN BEHAVIOR

RADA MIHALCEA

University of Michigan, mihalcea@umich.edu

Much of what we do today is centered around humans – whether it is creating the next generation smartphones, understanding interactions with social media platforms, or developing new mobility strategies. A better understanding of people can not only answer fundamental questions about “us” as humans, but can also facilitate the development of enhanced, personalized technologies. In this talk, I will overview the main challenges (and opportunities) faced by research on multimodal sensing of human behavior, and illustrate these challenges with projects conducted in the Language and Information Technologies lab at Michigan.

SCALING SEMANTIC ROLE LABELING AND SEMANTIC PARSING ACROSS LANGUAGES

ROBERTO NAVIGLI

Sapienza University of Rome, navigli@di.uniroma1.it

Sentence-level semantics is hampered by the lack of large-scale annotated data in non-English languages. In this talk I will focus on two key tasks aimed at enabling Natural Language Understanding, that is, Semantic Role Labeling and semantic parsing, and put forward innovative approaches which we developed to scale across several languages. I will show how new, language-independent techniques, as well as a brand-new, wide-coverage, multilingual verb frame resource, namely VerbAtlas, will help significantly close the gap between English and low-resource languages, and achieve the state of the art across the board.

CHAPTER 1.
LANGUAGE RESOURCES DEVELOPMENT, STANDARDIZATION AND EXPLOITATION

THE ROMANIAN MEDICAL TREEBANK - SIMONERO

VERGINICA BARBU MITITELU AND MARIA MITROFAN

Romanian Academy Research Institute for Artificial Intelligence

{vergi,maria}@racai.ro

Abstract

We present here the first Romanian medical treebank. It builds on a gold standard morphologically annotated corpus, also containing hand validated annotation with medical named entities. Enriched with a further linguistic level, namely syntax, it is a domain specific resource released within a multilingual context. We present quantitative data about it and the creation methodology. We also present and discuss here some comparative statistical data between this treebank and the general language treebank for Romanian.

Key words — corpus, medicine, named entities, Romanian, treebank, Universal Dependencies.

1. Introduction

Domain-specific language resources are valuable assets in natural language processing tools development and testing. We have been concerned with the medical domain for several years and have already reported two resources for it: a specialized corpus, BioRo (Mitrofan and Tufiș, 2018), created as part of the CoRoLa corpus (Tufiș *et al.* 2019), and a gold standard medical corpus (MoNERo) annotated morphologically (*i.e.*, PoS tagged) and with domain specific named entities (Mitrofan *et al.*, 2019). We continue here with the presentation of a new type of medical resource in Romanian, namely a newly released treebank, developed on top of MoNERo by adding a new annotation level, the syntactic one. It is thus called SiMoNERo.

A treebank is a corpus annotated at the syntactic level, with the tree as the representation of the sentence structure. Actually, the syntactic level of annotation is usually a further level of analysis in a treebank: on top of the morphological annotation, grammatical functions of words, dependencies between words, constituent boundaries become explicit.

Almost half a century ago manual annotation was the way to go to create treebanks – see Sampson (2003) who nostalgically remembers working on the first trees and being photographed from an aeroplane and then being sold the picture showing two disks close to each other in his yard, a pink one, his bald head, and a white one, the table covered with papers being drawn with trees. Nowadays, automatic annotation is the solution, although, Abeillé's 2003 remark that "human post-checking is always necessary" (Abeillé, 2003) still holds, as proven by the results of the Universal Dependencies Shared Task session within CoNLL 2018 (Zeman *et al.*, 2018), where the best ranked system had a Labeled Attachment Score of 75.84.

When developing a treebank, rarely do developers commit to a certain linguistic theory: Head-driven Phrase Structure Grammar (Simov *et al*., 2002; Oepen *et al.*, 2002) and Tree Adjoining Grammar (Shen and Joshi, 2005) are two of the few linguistic theories

reflected by existing treebanks. The decisions to make when taking up this endeavour are actually two: (i) choose between a dependency and a constituency annotation; (ii) choose between deep or shallow parsing, *i.e*. annotating only overt elements or also empty slots (Abeillé, 2003).

In this paper we present the creation of a new treebank for Romanian, a domain specific one, including medical texts, called SiMoNERo. Section 2 presents related work with respect to medical treebanks, on the one hand, and to Romanian treebanks, on the other hand. The preprocessing and annotation steps involved in the creation of this new treebank are presented in Section 3, while some statistics on it are given in Section 4, where we also draw a comparison between this treebank and a general language one, developed also within our group. We conclude the paper after we envisage potential uses of the treebank and offer information about how it can be accessed and queried in Section 5.

2. Related work

At present, there are treebanks available for tens of languages, most of them with an open license. Multiple treebanks for the same language are also available, developed by different authors, following different principles, adapted to domains, etc. One has to acknowledge the fact that this abundance of treebanks and their availability are also the results of the penetration of the Universal Dependencies[1] (UD) project principles and objectives in many research groups, of their dynamism and interest in the resources quality, especially in a multilingual context: only in the UD May 2020 release there were 163 treebanks for 92 languages, whereas in the UD November 2020 release 20 new treebanks were released and 12 new languages were represented.

2.1. Medical treebanks

As stated by Jiang *et al.* (2015), retraining existing parsers using medical treebanks is critical for improving their performance, while combining medical and general domain corpora can lead to achieving optimal performance for parsing clinical text. Therefore, several initiatives in the clinical NLP community have established the guidelines for annotating medical texts, as well as annotated corpora for parsing clinical text.

Fan *et al.* (2013) developed guidelines for parsing medical texts and annotated a corpus accordingly. They also created a treebank of 25 progress notes from University of Pittsburgh Medical Center. The annotated treebank contains 1,100 sentences, with a median length of 8 tokens per sentence, thus quite short ones.

Another annotated clinical corpus, named MiPACQ (Albright *et al.*, 2013), was created using pathology and other clinical notes from the Mayo Clinic. MiPACQ contains multiple layers of annotations, including named entities, syntactically parsed trees, dependency parsed trees and semantic role labeling on 13,091 sentences.

Within the UD project, several other treebanks, which also contain medical texts, were made public, but are not specific to this field, and the Romanian Reference Treebank

[1] universaldependencies.org

(see below) is one of them. For the Romanian language, SiMoNERo is the first medical treebank.

2.2. Romanian treebanks

To the best of our knowledge, there are several treebanks for Romanian available. The first created (Hristea and Popescu, 2003) was a dependency one, containing 4,042 short sentences selected from journalistic texts; its peculiarities are the analysis of clauses exclusively, not of sentences, and the exclusion of subordinating conjunctions from clauses; consequently, the language image it offers is deformed.

Another treebank is UAIC-RoDepTb (Perez, 2014), again a dependency one, containing 4,500 sentences, quite long ones, with an average of 37 words/sentence, manually annotated according to "traditional grammar" principles, while the list of relations also reflects the syntactic functions used in the Romanian traditional syntactic approach. The corpus is heterogeneous in structure, with texts from literature, from Romanian Wikipedia, from law texts, journalistic ones, etc. They are both original and translations.

Bick and Greavu (2010) report on a 21 million words journalistic treebank, automatically annotated within the Constraint Grammar formalism with a parser whose grammar is an adaptation of an Italian one.

Irimia and Barbu Mititelu (2015) created another dependency treebank (RACAI-RoTb) containing 5,000 sentences, semi-automatically annotated with a set of relations that was meant to be close to the UD one. The sentences were extracted from ROMBAC (Ion *et al.*, 2012), a balanced corpus of Romanian (reflecting the journalistic, medical, imaginative, juridical and scientific genres), and feature the most frequent verbs therein.

A conversion of UAIC-RoDepTb and RACAI-RoTb to UD format resulted in a reference treebank for Romanian (RoRefTrees or RRT) (Barbu Mititelu *et al*., 2016) which was released in UD. It contains 9,523 sentences, covers a variety of genres, reflects the contemporary language and was manually validated at the syntactic level.

Another treebank for Romanian released within UD is Romanian Non-standard or UAIC-RoDia (Colhon *et al.*, 2017). With its 572,436 tokens, it is the largest available here. It stands out due to the fact that it contains texts from older periods of the language (16th to 19th centuries), as well as from folklore.

3. Treebank description

In this section we present the corpus content from the texts types perspective, its processing and the levels of annotation available.

3.1. Types of texts in the corpus

SiMoNERo consists of texts extracted from three types of documents: medical scientific journal articles, scientific medical literature books and medical blog posts, but most of them are those coming from medical books. The main reason for choosing these three sources was the good quality of the texts, the correct usage of medical terminology and the abundance of medical terms. All the sentences were extracted based on the metadata

scheme associated with each document present in BioRo, the corpus from which SiMoNERo was extracted. All texts are I(ntellectual)P(roperty)R(ights)-cleared, which is a valuable asset in the perspective of offering large access to the resource.

3.2. Levels of annotation

The texts were sentence split, tokenized and lemmatized using the TTL tool (Ion, 2007). The annotation scheme has three different levels:

i) *the morfologic level* was developed in two steps: automatic annotation performed with the TTL tool (Ion, 2007) followed by the manual verification of the tags. During this phase, several types of errors were corrected (Mitrofan *et al*., 2018). The annotation scheme used was based on the MSD tag-set developed in the Multext-East project (Dimitrova *et al*., 1998), which contains 715 tags for Romanian and fourteen classes of words.

ii) *the named entity level* was manually developed by two annotators: one physician and one experienced annotator, both having Romanian as native language. The annotation scheme of the named entities (NE) was based on four UMLS[2] semantic groups: anatomy (ANAT), chemicals and drugs (CHEM), disorders (DISO) and procedures (PROC). The main reason for choosing these four types of entities was a trade-off between the minimum number of entities of each type and the maximum relevance for our corpus. Since the corpus was tokenized and in CONLL-U[3] format, the IOB2 (Inside-Outside-Beginning) (Sang and Veenstra, 1999) format was chosen to represent the named entities. The B-tag is used for the first token (so the beginning) of every NE, the I-tag indicates the token that is inside an NE and the O-tag is used for surrounding tokens that do not belong to an NE.

iii) *the syntactic level* was automatically added using the NLP-Cube parser[4] (Boroș *et al.*, 2018) that was trained on RRT. A validation process was run so as to ensure the treebank's conformance with the UD specifications: a lot of manual intervention was necessary so that all validation tests[5] created in UD are now passed by SiMoNERo. One such example is represented by the removal of auxiliary chains from annotations: *e.g.* in Figure 1, we show how the auxiliary *ar* must be attached to the head of the clause, the adjective *utilă*, and not to the verb *fi*, because the latter's part of speech is also `AUX` (given its copula reading here); this type of annotation goes against the morphological knowledge: the auxiliary *ar* is used here for creating the present conditional of the verb, so a certain mood, which is a grammatical category of the verb, not of the adjective; however, it is the copula reading of the verb *fi* which prevents such annotation in this kind of examples: the adjective is considered the head of the clause and all functional categories related to the predicate (auxiliaries among them) depend on it.

[2] https://semanticnetwork.nlm.nih.gov/
[3] https://universaldependencies.org/format.html
[4] https://opensource.adobe.com/NLP-Cube/index.html
[5] A comprehensive list of these tests is available at https://universaldependencies.org/svalidation.html.

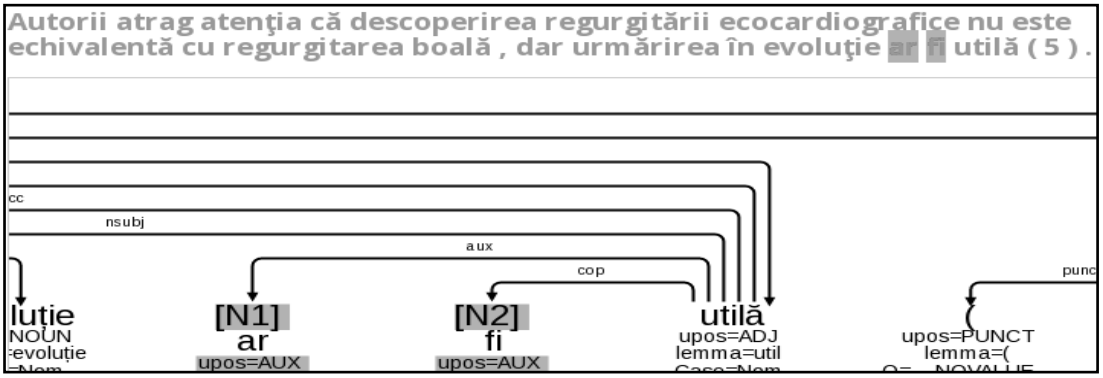


Figure 1: Avoiding chains of auxiliaries

4. Statistics of the treebank

4.1. SiMoNERo

In this section we present general statistics about the treebank. Table 1 shows the distribution of sentences within the medical domains. It can be seen that this distribution is not balanced because of the copyright restrictions which made it impossible to collect the same amount of texts for each medical domain.

Table 1: Distribution of texts from medical domains in the corpus

Domain	**Tokens**	**Sentences**
Cardiology	40.7%	40.6%
Diabetes	44.7%	43%
Endocrinology	14.6%	16.4%

The treebank contains 4,681 sentences split into three files, as shown in Table 2 (see also Section 5), where we included the average length of sentences in each file, which is always at least 30 tokens/sentence. The bottom line of the table includes similar information about the medical component of RRT: we notice that the sentences here are shorter that the ones in SiMoNERo.

Table 2: Number of sentences and their average length. Comparison with the medical subcorpus of RRT

File	**Number of sentences**	**Average sentence length (tokens/sentence)**
SiMoNERo train	3,747	31
SiMoNERo dev	443	33
SiMoNERo test	491	30
medical RRT	1,210	23

Analysing the content words, it turned out that the texts have a descriptive structure, 27.8% of content words being nouns, followed by adjectives, 11.5%, and 10.4% verbs. The multitude of cases in which nouns are followed by two or more adjectives contributes to the descriptive character of the texts.

SiMoNERo also has 14,133 medical named entities marked among the tokens, distributed in the four types mentioned above as shown in Table 3. As expected, medical texts mainly describe diseases and medical conditions, that is why NEs belonging to the DISO semantic group are prevalent.

Table 3: Types of medical named entities and their number in SiMoNERo

Type	Number
DISO	6,611
CHEM	4,156
ANAT	1,964
PROC	1,402

4.2. SiMoNERo versus RRT

In this section we compare the two Romanian treebanks reflecting contemporary language that are available in UD, namely SiMoNERo, as a medical treebank, and RRT, as a general language treebank, with the aim of highlighting the characteristics of the former.

SiMoNERo contains much longer sentences, as well as a higher frequency of punctuation signs: there are a lot of scientific data in the form of results of measurements or analyses (*e.g. 180 mg / dl* or *TA < 120 / 80 mmHg*; there are 401 slashes in SiMoNERo) rendered also as percentages (*e.g. 77% dintre pacienți* "77% of the patients") and as intervals of values (*e.g. TA diasistolice 90 - 99 mmHg*). Moreover, many statements are backed by references to papers, rendered in the form of numbers between brackets sending to a position in the list of references (*e.g.* (3) or [3]), while further explanations are given also between brackets within the sentence (*e.g. substituție C - T silențioasă (care nu determină modificarea aminoacidului codificat)* "silent C - T substitution (which does not determine the modification of the coded amino acid)"); there are 2,540 pairs of brackets in SiMoNERo and only 1,671 pairs in RRT.

Besides their different size, Table 4 also shows the lexical diversity of each treebank: we can see that both of them are characterized by lexical repetitiveness: the percent of unique lemmas is below 10 in both treebanks. However, comparing the vocabulary specific to each of them we notice that 58% of the unique lemmas in SiMoNERo do not occur in RRT. They are mainly medical terms: *infarct* (heart attack), *reparatoriu* (reparatory), *compresiv* (compressive), *cefalalgic* (cephalgic), *neuroglicopenic* (neuroglycopenic), *toracotomie* (thoracotomie), *osteoblastic* (osteoblastic) etc. There is a higher percentage of lemmas specific to RRT, namely 74%, and this can be explained by the multiple domains that coexist in it: law, sciences, literature, journalism, medicine, wikipedia etc. Some examples are: *peștișor* (little fish), *împrejur* (around), *paralelipiped* (parallelipiped), *alocat* (allocated), *zdrențuit* (ragged), *înger* (angel), *turnesol* (litmus), *behăială* (bleat), *pneu* (tyre), *arbitru* (referee), *contribuabil* (taxpayer), *penumbră* (penumbra), *urologic* (urologic), *frescă* (fresco), *pricepere* (know-how), *neorânduială* (disorder), *interstelar* (interstellar), *comerciant* (trader), *sat*

(village), *cutremur* (earthquake), *sfoară* (rope), *artă* (art), *tehnician* (tehnician), *necinste* (dishonesty), *zevzec* (fool).

As expected, proper nouns are more numerous in RRT (17% of the unique lemmas) than in SiMoNERo (only 4% of the unique lemmas). Nouns (proper ones excluded) are rather equally represented in the two resources (47% in SiMoNERo and 45% in RRT). However, SiMoNERo's descriptive character mentioned above is supported by the higher frequency of adjectives, both when considered as unique lemmas and when considering their actual occurrence in the corpus.

Table 4: General data - a comparison between SiMoNERo and RRT

	SiMoNERo	**RRT**
sentences	4,681	9,523
tokens	146,020	218,511
tokens / sentence	31.19	22.94
punctuation	19,614	27,506
punctuation / sentence	4.2	2.9
adjectives	17,053	15,229
unique lemmas	10,711	17,458
unique lemmas minus proper nouns	10,282	14,409
unique lemmas minus proper nouns, numerals and punctuation	9,346	13,456
unique lemmas - only nouns	5,012	7,925
unique lemmas - only adjectives	2,891	3,484
% of adjectives	12	7
% of unique lemmas	7	8
% of proper nouns in unique lemmas	4	17
% of unique lemmas minus proper nouns, numerals and punctuation	87	77
% of numerals and punctuation	9	6
% of nouns in unique lemmas	47	45
% of adjectives in unique lemmas	27	20
lemmas only in SiMoNERo	6,210	-
lemmas only in RRT	-	12,957
% lemmas only in one treebank	58	74

5. Format, Access, Query and Use of SiMoNERo

SiMoNERo is UTF-8 encoded, with LF character as line break and is available in CONLL-U format, having named entities marked on the last column.

It was split in a random fashion in three files as follows: the test set (ro_simonero-ud-test.conllu) is 10% of the whole treebank, the development set (ro_simonero-ud-dev.conllu) also 10%, while the rest of the treebank (80%) is the training set (ro_simonero-ud-train.conllu) (see also Table 2). The whole treebank is freely available for download[6] under a CC BY-SA 4.0 license.

[6] https://github.com/UniversalDependencies/UD_Romanian-SiMoNERo/find/master

The treebank is available for querying online, alongside other treebanks, on two platforms: PML Tree Query[7] and Grew-match[8].

Being the first Romanian biomedical treebank annotated with both part of speech tags and named entities, SiMoNERo has an important contribution in named entity recognition (Ion *et al*., 2019), machine translation (Neves *et al*., 2018) and other NLP tasks. In order to render it more importance, we intend to proceed to a systematic improvement of the syntactic annotation for the next UD release.

6. Conclusions

The Romanian medical treebank SiMoNERo is the outcome of our interest in developing medical language resources. It was the next natural step after releasing MoNERo, the gold standard morphologically annotated corpus, with domain specific named entities (Mitrofan *et al.*, 2019). In a larger context, it is in line with our concern with the development of language resources in general and, in an even larger context, with the community's understanding of their importance for the development of language processing tools. Further, we intend to annotate the NEs with semantic information such as WordNet senses and also to annotate the events. The treebank will also be made available in a standardized format, namely linguistic linked open data.

References

Abeillé, A. (2003). *Treebanks. Building and Using Parsed Corpora*, Dordrecht, Boston, London, Kluwer Academic Publishers.

Albright, D., Lanfranchi, A., Fredriksen, A., Styler IV, W.F., Warner, C., Hwang, J.D., Choi, J.D., Dligach, D., Nielsen, R.D., Martin, J., Ward, W. (2013). Towards comprehensive syntactic and semantic annotations of the clinical narrative. *Journal of the American Medical Informatics Association, 20*(5), 922-930.

Barbu Mititelu, V., Ion, R., Simionescu, R., Irimia, E., and Perez, C.-A. (2016). The Romanian Treebank Annotated According to Universal Dependencies. *Proceedings of HrTAL2016*, Dubrovnik, Croatia, 29 September - 1 October 2016.

Bick, E. and Greavu, A. (2010). A Grammatically Annotated Corpus of Romanian Business Texts. In *Proceedings of Multilinguality and Interoperability in Language Processing with Emphasis on Romanian*, Editura Academiei Romane, 169-183.

Boroș, T., Dumitrescu, S. D. and Burtica, R. (2018). NLP-Cube: End-to-End Raw Text Processing With Neural Networks. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Association for Computational Linguistics, 171-179.

Colhon, M., Mărănduc, C. and Mititelu, C. (2017). A Multiform Balanced Dependency Treebank for Romanian. In *Proceedings of Knowledge Resources for the Socio-Economic Sciences and Humanities, (KnowRSH)*, Varna, Bulgaria, September 8,

[7] http://lindat.mff.cuni.cz/services/pmltq/#!/home
[8] http://match.grew.fr/

2017 workshop at the Recent Advances in Natural Language Processing (RANLP), 9-19.

Dimitrova, L., Ide, N., Petkevic, V., Erjavec, T., Kaalep, H.J. and Tufiș, D. (1998). Multext-east: Parallel and comparable corpora and lexicons for six central and eastern european languages. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98/COLING '98, 315-319, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fan, J.W., Yang, E.W., Jiang, M., Prasad, R., Loomis, R.M., Zisook, D.S., Denny, J.C., Xu, H. and Huang, Y. (2013). Syntactic parsing of clinical text: guideline and corpus development with handling ill-formed sentences. *Journal of the American Medical Informatics Association*, *20*(6), 1168-1177.

Hristea, F., Popescu, M. (2003). A Dependency Grammar Approach to Syntactic Analysis with Special Reference to Romanian. In F. Hristea, M. Popescu (eds), *Building Awareness in Language Technology*, București, Editura Universității din București, 9-16.

Ion, R. (2007). *TTL: A portable framework for tokenization, tagging and lemmatization of large corpora*, PhD dissertation, Romanian Academy, Bucharest (in Romanian).

Ion, R., Irimia, E., Ștefănescu, D. and Tufiș, D. (2012). ROMBAC: The Romanian Balanced Annotated Corpus. In *Proceedings of LREC 2012*, Istanbul, Turkey, 339-344.

Ion, R., Păiș, V.F. and Mitrofan, M. (2019). RACAI's System at PharmaCoNER 2019. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, 90-99.

Irimia, E., Barbu Mititelu, V. (2015). RACAI-RoTb: nucleu de corpus de limbă română adnotat sintactic cu relaţii de dependenţă, *Revista Română de Interacţiune Om-Calculator* 8 (2) 2015, 101-120.

Jiang, M., Huang, Y., Fan, J.W., Tang, B., Denny, J., Xu, H. (2015). Parsing clinical text: how good are the state-of-the-art parsers?. *BMC medical informatics and decision making*, *15*(S1), p.S2.

Mitrofan, M., and Tufiș, D. (2018). BioRo: The biomedical corpus for the Romanian language. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, 1192-1196.

Mitrofan, M., Barbu Mititelu, V., Mitrofan, G. (2018). Towards the Construction of a Gold Standard Biomedical Corpus for the Romanian Language. *Data* 3.4 (2018): 53; https://doi.org/10.3390/data3040053.

Mitrofan, M., Barbu Mititelu, V. and Mitrofan., G. (2019). MoNERo: A Biomedical Gold Standard Corpus for the Romanian Language. In *Proceedings of the 18th BioNLP Workshop and Shared Task, ACL*, 71-79.

Neves, M., Yepes, A.J., Névéol, A., Grozea, C., Siu, A., Kittner, M. and Verspoor, K. (2018). Findings of the wmt 2018 biomedical translation shared task: Evaluation on medline test sets. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, 324-339.

Oepen, S., Flickinger, D., Toutanova, K. and Manning, C. D. (2002). LinGORedwoods. A rich and dynamic treebank for HPSG. In *Proceedings of the 1st Workshop on Treebanks and Linguistic Theories*, 139-149.

Perez, C. A. (2014). *Linguistic Resources for Natural Language Processing*, PhD dissertation, A.I. Cuza University of Iasi (in Romanian).

Sampson, G. (2003). Thoughts on Two Decades of Drawing Trees. In Abeillé, A. (ed.) *Treebanks. Building and Using Parsed Corpora*, Dordrecht, Boston, London, Kluwer Academic Publishers, 23-42.

Sang, E. F. and Veenstra, J. (1999). Representing text chunks. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, 173–179. Association for Computational Linguistics.

Simov, K., Osenova, P., Slavcheva, M., Kolkovska, S., Balabanova, E., Doikoff, D., Ivanova, K., Simov, E. and Kouylekov, M. (2002). Building a Linguistically Interpreted Corpus of Bulgarian: the BulTreeBank. In *Proceedings of LREC 2002*, Canary Islands, Spain, 1729-1736.

Shen, L. and Joshi, A. K. (2005). Building an LTAG treebank. Technical Report MS-CIS-05-15, CISDept., UPenn.

Tufiș, D., Barbu Mititelu, V., Irimia, E., Păiș, V., Ion, R., Diewald, N., Mitrofan, M. and Onofrei, M. (2019). Little strokes fell great oaks. Creating CoRoLa, the reference corpus of contemporary Romanian. *RRL*, LXIV, 3, 227-240.

Zeman, D., Hajic, J., Popel, M., Potthast, M., Straka, M., Ginter, F., Nivre, J. and Petrov, S. (2018). CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, October, Brussels, Belgium, ACL, 1-21.

PARSING TEMPORAL AND SPATIAL INFORMATION

CĂTĂLINA MĂRĂNDUC[1], VICTORIA BOBICEV[2], AND CENEL AUGUSTO PEREZ[1]

[1] Faculty of Computer Science, Al. I. Cuza University, Iași
[2] Technical University of Moldova, Chișinău
catalinamaranduc@gmail.com, victoria.bobicev@ia.utm.md, augusto.perez@info.uaic.ro

Abstract

In this paper we present a dependency treebank morphologically and syntactically annotated in a specific scheme. We managed to increase the accuracy of the POS-tagger and the syntactic parser used, which led to the increase in the volume of annotated texts. First, we analysed the accuracy with which the syntactic parser recognizes the 14 types of circumstantial complements, especially the temporal and spatial ones. These are the most numerous circumstantial complements, and they are very important for the configuration of a textual world describing reality or proposing a fictitious world, providing information about the type of text. In December 2020 our treebank comprised 42,542 sentences (919,608 words and punctuation). We studied our documents containing fictional and non-fictional narrative. Using a Malt parser optimizer, we extracted dependency chains of time and spatial complements. The number of complements and the degree to which they are precise is related to the type of text, fictional or nonfictional. In order to construct a classifier of texts, one can count the spatial and temporal complements and one can observe if they represent determinations of exact landmarks (with geographical proper names and numbers) - in which case the text is a real narrative, or if they represent imprecise determinations, in which case the narrative is fictional.

Key words — Local complements, narrative fiction, narrative reality, syntactic parser, temporal complement, treebank, type of text.

1. Introduction

The RoDia (Romanian Diachronic) Dependency Treebank was created in 2007 and it increased to 4,600 sentences in 2014 (Perez, 2014). Regarding the basic syntactic format, created in 2007 in accordance with the Dependency Grammar principles (Tesnière, 1959; Mel'Čuk, 1987), we have only made insignificant changes since 2014. The list of complements includes 14 types of circumstances and the coordination is a chain starting from the first coordinate, which also includes the connecting words or punctuation. But the volume of the treebank has increased a lot with the improvement of automatic annotation tools. We have used a hybrid POS-tagger (Simionescu, 2011), which we adapted for Nonstandard Romanian, the list of morphological labels from the MULTEXT-East project (Erjavec, 2012), as well as diverse variants of the MaltParser (Hall *et al*., 2006), trained on the growing gold corpus that we created. The treebank is corrected manually, but this is getting easier as the number of errors decreases. In December 2020, it comprises 42,542 sentences, with 919,608 words and punctuation.

We have focused on old Romanian texts from the 16th-19th centuries, and we have annotated whole books, because we have noticed that in this way the parser is better trained on more and more specific structures, and the texts are also available for other types of research.

In November 2017, a treebank for Nonstandard Romanian was created on the Universal Dependencies (UD) portal (Mărănduc and Bobicev, 2017). After three years, in November 2020, the treebank is available with 26,221 sentences (572,259 tokens, punctuation included).

Regarding the semantic annotation convention, we failed to create a semantic parser for it and that is why the semantic treebank has only 5,566 sentences with 99,341 tokens (Mărănduc *et al.*, 2018). As a first step towards creating a semantic parser, we have tried to train diverse Malt Parser variants (Smith *et al.*, 2018) available on the UD site, on the basic format of our treebank, with the 14 types of circumstantial complements, attempting to improve their accuracy.

The transformation into the UD convention is done automatically using the Treeops program (Colhon *et al.*, 2017) and the result depends on the correctness of the morphological and syntactic annotation in the basic format.

A method for obtaining a better accuracy is to increase the training corpus with texts rich in the type of complement to which poor accuracy is recorded, because it has too few attestations in the texts. Thus, in order to study the time-space confusions made by the parser, we annotated Neculce's chronicle, which is very rich in spatial complements and quite rich in time complements, being a non-fiction narrative text.

The automatic parser mistakes certain pairs of circumstantial complements:

- time and place;
- associative and instrumental;
- conditional and concessive;
- conditional and consecutive;
- cause and purpose.

We refer here only to the local and temporal ones, which are very frequent and very important for the configuration of the textual world, be it fictional or non-fictional, and of the text type. If we solve the annotation of one of the 2 categories, we also solved the one with which it is confused. If we manage to solve the correct annotation of time, then all other information that refers to landmarks or directions or sequences is spatial.

Time annotation of circumstantial complements is preserved in the UD format by the existence of three sub-classifications specific for Romanian treebanks: nmod:tmod, advmod:tmod, advcl:tcl.

To increase the accuracy of complement recognition, we have used several methods, including re-correcting those types of complements where we found a large number of errors, based on inconsistencies in the training corpus.

For example, if the erroneous annotation of the constructions: *pe vremea aceea*, *pe cea vreme*, *în vremea ceea* (En: *at that time*, *at the time mentioned*) as being c.c.l. (space

complement) was repeatedly found in the automatic parsing, then we looked for the word *vremea* (En: *time*) in the training corpus to discover further errors not corrected and that generated the current errors. In the first of the five sub-corpora, Neculce's "Cronicle", we found 4 errors. These generated 10 errors in the second sub-corpus and an even higher number of errors in the others. The issue would be solved if all of these errors were corrected.

Thus, the lexical elements that should induce the annotation of the dependency relationship as temporal are extracted: *an*, (archaic synonyms: *leat, let, vleat*), *zi, lună, dimineața, seara, noapte, ianuarie, februarie, martie, aprilie, mai, iunie, iulie, august, septembrie, octombrie, noiembrie, decembrie, luni, marți, miercuri, joi, vineri, sâmbătă, duminică, iarnă, vară, primavară, toamnă, timp, anotimp, vreme, săptămână, oră, zi, veac, veci, început, sfârșit, când, cândva, oricând, atunci, acum (acu, acuș), mereu, totdeauna, apoi (păi), mâine, poimâine, ieri, alaltăieri, odată, diseară, târziu, devreme, astăzi, (azi), imediat, deocamdată, curând, pururea* (En: *year, day, month, morning, evening, night, January, February, March, April, May, June, July, August, September, October, November, December, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday, winter, summer, spring, autumn, time, season, weather, week, hour, day, century, forever, beginning, end, when, sometime, anytime, then, now, always, all the times, then, tomorrow, the day after tomorrow, yesterday, the day before yesterday, once, tonight, late, early, today, immediately, for now, soon, forever).*

In order for the parser to memorize them as inducers of the c.c.t. relationship, each of them must be annotated with that relationship at all occurrences with this temporal meaning in the treebank, without any error disturbing the induction process. This is the way for the parser to memorize the words related to the notion of time, following the training with correctly annotated texts.

A solution could also be to link the words in the treebank to the information in the Romanian WordNet, as was done recently with a parsing experiment with which an increase in accuracy of 0.5 percent was obtained (Barbu Mititelu *et al.*, 2016).

2. Parsing Experiments

We parsed our documents with MaltParser (Smith *et al*., 2018), a data-driven parser. This parser demonstrated the ability to obtain good results for multiple languages and has been widely used in Universal Dependencies projects (Nivre *et al*., 2016). The first set of experiments was performed with the whole corpus. The whole corpus, except for one document, was used for training, and testing was performed on the document that was excluded from the training set, thus obtaining the data presented in Table 2.

However, the documents in our corpus are quite different and training on one of them and testing on other results in poor accuracy. Thus, we decided to experiment with every document separately. Some of them are relatively small, too small to be used for training, and we selected the three largest documents in our corpus, namely the New Testament *Gospels* and *Acts*, and Neculce's *Chronicle*.

The MaltParser offers a wide range of parameters for optimization, including nine different parsing algorithms, two different machine learning libraries (each with a number of different learners), and an expressive specification language that can be used

to define arbitrarily rich feature models. In our case we are especially interested in the feature set optimization.

First of all, we used the MaltOptimizer (Ballesteros and Nivre, 2012) to detect the best algorithm and feature set for our documents. The MaltOptimizer processed the documents in three steps. The first step was used to gather information about the various properties of the training set. During the second step, the MaltOptimizer explored a subset of the parsing algorithms implemented in the MaltParser, based on the results of the data analysis to detect the best one for this particular training set. The goal of the third step was optimization of the feature model given by the parsing algorithm chosen. It tested potentially useful features one by one and in combination to ensure that all features in the model actually make a contribution. The result of MaltOptimizer use is presented in Table 1.

Table 1: The best algorithms and the best Labelled Attachment Score (LAS) for three largest documents of our corpus.

Document	**Best Algorithm**	**Best Performance (LAS)**
New Testament Gospels	Nivreeager	83,8
New Testament Acts of Apostles	Nivrestandard	78,9
Neculce Chronicle	Nivreeager	84,59

Most of the effort when optimizing MaltParser usually goes into feature selection, that is, in tuning the feature representation that constitutes the input to the classifier. A feature model in MaltParser is defined by a feature specification file in XML. It states that the parsing algorithm uses 32 features including: POSTAG values of the neighbouring tokens around the current token; 4 FORM that presents words around the current token; LEMMA of the current token, 3 DEPREL (dependency relation labels) and 8 complex features that merge two or three features as, for example, morphological label of the word and its dependency relation to the left and to the right.

3. Related Work

The annotation of space and time, as a means of configuring textual worlds or communication situations, is increasingly in the attention of linguists and computer scientists. It is also the basis for the search for time information retrieval, TIR, or geographic information retrieval, GIR. Strötgen (2010) shows how co-occurrences of spatial and temporal information are determinant for the spatio-temporal profiles of documents.

Llorens *et al.* (2009) only deal with the annotation of temporal semantic roles, in accordance with the internationally accepted TimeML scheme, and evaluates a set of time-related MWEs, TIMEX3 in English and Spanish, with an accuracy of 76%, which makes the authors consider that they are likely to be identified in other languages as well. Three years later, Llorens *et al.* (2012) propose an automatic system for identifying time relationships in natural language. The experiments were made on an available English data set annotated with temporal information (TimeBank) in a 10-fold cross-validated evaluation, with an accuracy of 46%.

In the paper (Lefeuvre *et al.*, 2016), a syntactic rather than lexical annotation of time in a treebank in French is described, and the authors make proposals to extend the TimeML scheme. An annotation of temporal dependency structure is performed on a corpus of children's narratives in (Kolomiyets *et al.*, 2012). The agreement among more annotators is: 0.856 on the event words, 0.822 on the links between events, and of 0.700 on the ordering relation labels.

In Romanian, the English corpus of Time Bank was ported in Romanian (by translation) with all temporal annotations (Forăscu and Tufiș, 2012), having 4715 sentences (65,375 tokens). A conference of the same year, on semantic web data annotation focuses, among other things, on the recognition of TimeML noun events, *i.e.* on a scheme for processing the event and temporal expressions in natural language processing fields (Jeong and Myaeng, 2012).

A chapter in a Springer book is also interested in a database which can manage events that are evolving with time, *i.e.*, the information of spatial objects whose shape and position evolve with time (Xiaoping *et al.*, 2011).

Our corpus consists of texts in Old Romanian and is not annotated with the categories in Time Bank, but it could be because all the information about the modes and tenses are in X-Postag (the morphological annotation specific to our treebank).

Therefore, we did not give up the annotation of the verbal circumstantial (time) modifiers in our basic syntactic convention; we have tried to see what information we can extract from the syntactically annotated corpus we hold. In the semantic format, we managed to annotate the space and time when they are verbal or nominal determinants, but in this paper we discuss only verbal modifiers.

4. Narrative Corpus Content

Table 2 below presents the documents included in this study, annotated morphologically and syntactically, the number of sentences and tokens, the accuracy of the automatic parser (labelled and unlabelled attachment score, *i.e.* LAS and UAS) and the type of the texts. It is a balanced corpus, *i.e.* the contemporary and the old texts, the regional ones, and the social media communication are all represented. The first word in the title, with capitals, marks these categories. For the old texts, the century is also added. We excluded from this study the legal style, Wikipedia, the lyrical poetry, popular and church (Psalms). For the classification of these types of texts, we need some other criteria (Mărănduc, 2005).

The information on the time and space framing appears in narrative texts. We can study the number of such complements and what would be their form when the narrative is fictional, compared to when the narrative is mystical or a reality one. These complements are also found in dialogues, where they circumscribe the communication situation.

Table 2: The files included in this study

Name of xml	Sent.	Tokens	LAS	UAS	type of text
CHAT	2,579	39,239	82.89	73.57	dialogue narrative reality
CONT_1984_orwell	904	17,608	82.05	73.28	novel narrative fiction
CONT FrameNet	1,092	24,659	79.33	71.10	journal narrative reality
OLD XVI Flower of Gifts	1,083	21,078	82.95	74.68	philos narrative fiction
OLD XVII NewTest. Gospel	5,174	92,440	85.39	79.07	church narrative dialogue
OLD XVII NewTest. Apostles	5,901	122,290	84.50	77.13	church narrative epistolary
OLD XVIII Neculce Chronicle	6,068	157,694	89.16	84.78	chronicle narrative reality
OLD XIX Caragiale Kings +Remember	1,691	48,930	77.09	70.09	novel narrative fiction
POP Ballads Rep Mold	1,112	19.405	80.04	72.37	narrative fiction
POP Ballads Rom	1,486	36,780	79.08	72.64	narrative fiction

Table 3 shows first the total number of occurrences for the principal circumstantial complements in each document, secondly the correctly parsed ones, and thirdly the erroneous occurrences, manually corrected. The temporal and spatial complements are mistaken for each other, the local annotated as temporal being less numerous than the temporal ones annotated as being local. Correctly annotated complements do not exceed half of the total number of occurrences, except in the case of local and modal ones. The table shows that the syntactic parser cannot yet correctly annotate semantic categories of information and semantic relationships.

Table 3: Occurrences of circumstantial complements in the documents, the total number, the correct parsed and the erroneous corrected relations.

c.c.	chat	Neculce	Frame Net	Gospel	Apostle	Flower	Orwell	Carag	Ballads RM	Ballads RO
c.c.l.	983 668 315	6695 4668 2027	558 433 125	2685 1746 939	2921 2120 801	441 350 91	468 332 136	1422 1167 255	1313 873 440	553 343 210
c.c.t.	1216 407 809	3161 1373 1788	763 368 395	1568 747 821	1540 754 786	311 171 140	417 184 233	1105 696 409	490 208 282	234 96 138
c.c.m.	1791 1168 623	4688 2781 1907	816 430 386	2013 906 1157	3628 1489 2139	496 282 214	982 575 407	2165 1572 693	757 427 330	517 316 201
c.c.cz.	328 35 293	1561 693 868	162 34 128	91 47 44	1487 720 767	226 111 115	63 33 30	304 135 169	338 189 149	132 96 36
c.c.	294	1577	99	67	1053	166	40	277	337	144

scop	68 226	684 893	54 45	28 39	481 572	92 74	13 27	157 120	168 169	47 97
c.c. cond.	138 80 58	639 177 462	74 43 31	34 15 18	486 235 251	136 60 76	26 21 5	105 59 46	76 37 39	57 28 29
c.c. conc.	39 1 38	104 11 93	51 3 48	79 4 75	117 6 111	10 0 10	39 6 33	104 62 42	27 1 26	22 1 21
c.c. cons.	36 5 31	1265 659 606	59 21 38	10 28 74	188 34 154	55 26 29	32 17 15	90 25 65	42 14 28	29 9 20
c.c. instr.	112 43 69	572 340 232	32 19 13	26 12 13	518 228 290	53 36 17	33 17 16	203 126 77	166 51 115	123 49 74
c.c. soc.	103 59 44	1112 441 671	23 11 12	31 18 12	407 232 175	65 23 42	29 15 14	199 143 56	78 23 55	43 15 28

5. Morphological Classification of Temporal and Spatial Circumstantial Complements

Circumstantial complements are generally considered modifiers, which are not part of core dependencies. However, there are certain verbs that by their meaning have such dependencies as necessary and mandatory. An approach related to the realization of dependency relationships through various formal structures can be found in (Bejček *et al.*, 2018), which presents a resource called Forms and Function (ForFun) and the usage possibilities. They present an inventory of the multiple forms that spatial and temporal complements can have and how the preferred selection of some of them can give useful information for others studies. For example, in the classification of texts, a poem that uses narrative or dialogue to highlight the lyric could be classified as lyrical. If we have a narrative that uses the convention of the found notebook, we could classify it as fictional narrative.

5.1. Adverbs

Regarding adverbs, those included in the list in Section1 can be considered as specific to the temporal relationship, but many others can appear with this relationship although they are non-specific. For example, *cum* (En: *how*), which is specific to the circumstantial modality relation, can have a temporal value, as in Ex 1:

Example 1:

Cum s-a culcat, a și început a sforăi. 'How Refl.Cl.-has lain_down, has even begun to snore' (En: *The moment he lay down, he began to snore.*)

Other adverbs are ambiguous, because though they express a succession, it can be a spatial or temporal one. For example, the adverb *înainte* (En: *before*) is frequently both spatial and temporal. In the first example, it has a local meaning, and in the second example, it has a temporal one (see Fig. 1 and 2).

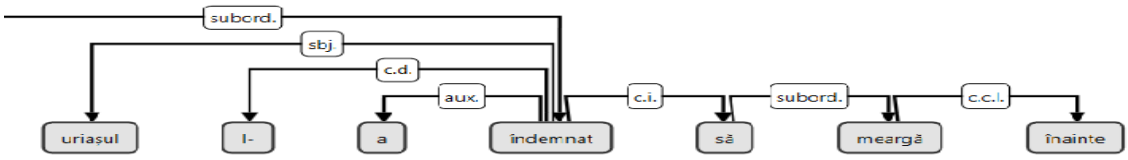


Figure 1. *Înainte* as spatial adverb. *Uriașul l-a îndemnat să meargă înainte*. En: *The giant urged him to go ahead.*

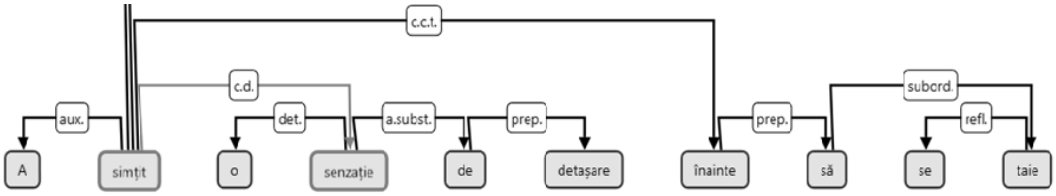


Figure 2. *Înainte* as temporal adverb. *A simțit o senzație de detașare înainte să se taie*. En: *He felt a sensation of detachment before cutting himself.*

If they are in relation to other adverbs that are specialized strictly as adverbs expressing temporal relations, the type of their relation will be disambiguated in accordance with their relation.

5.2. Nouns, Pronouns and Adverbs Preceded by Adpositions

Any of the words in the previous lists can be found with any of the adpositions, and being in circumstantial temporal relation, although some adpositions are more frequently used for a particular relationship, for example *pe*, *spre către* (En: *on*, *to*, *towards*) are used more in local complements than in temporal ones.

Certain word groups are usually time complements: *până ieri, până la anu, de mâine, pe urmă, în acel an, după aceea, după ce,* etc. (En: *until yesterday, until next year, as of tomorrow, afterwards, that year, afterwards, after* etc.)

Also, an adposition followed by a noun with a cardinal or ordinal numeral is a circumstantial complement of time.

5.3. Clauses with Temporal and Spatial Information

Clauses that fulfil the spatial or temporal modifier relation, as well as the words accompanied by prepositions, can select typical or common introductory elements with other types of relations. They show an action or state that is in a relationship of spatial or temporal succession to the action or state in the regent clause.

A large number of temporal clauses are introduced by relative pronouns with adpositions: *după ce, după care, până ce* (En: *after, after which, until*).

6. Types of Local and Temporal Complements Specific to Text Types

Table 4 shows the relationship among the number of sentences in a document, the type of text, and the number of temporal and local complements. In the case of temporal

complements, the precision of the determination is achieved by the existence of a number, whether it is a cardinal or ordinal numeral. In the case of local complements, the precision of the determination is achieved by the existence of a proper noun in the word chain, especially if it is a geographical name.

If we build a program that classifies into text types morpho-syntactically annotated texts entered into a database, we should enable it to look for certain parameters, such as the ratio of the number of sentences to the number of time modifiers. If the number of time complements were less than half the number of sentences, then we would have a fictional narrative text or a lyrical text. The value of 0.5 that we provisionally give this parameter can be found if we evaluate the same number of texts whose text type is already established.

Table 4: Correspondence among the type of the text, the number of temporal and local complements, with/without numerals and proper names

Name of xml	**Sent.**	**c.c.t.**	**c.c.t. with NUM**	**c.c.l.**	**c.c.l. with Np**	**type of text**
CHAT	2,579	1216	103	983	94	dialogue
CONT 1984 Orwel	904	417	9	468	10	novel narrative fiction
CONT Frame Net	1,092	763	38	558	81	journal narrative reality
OLD XVI Flower of Gifts	1,083	312	4	441	34	philosophy narrative fiction
OLD XVII New Test. Gospel	5,174	1,568	44	2,685	269	church narrative dialogue
OLD XVII New Test. Apostles	5,901	1,540	59	2,921	527	church narrative epistolary
OLD XVIII Neculce Chronicle	6,068	3,158	144	6,696	2129	chronicle narrative reality
OLD XIX Caragiale Princes	1,691	939	14	1,190	133	novel narrative fiction
POP Ballads Rep Mold	1,112	317	4	717	14	narrative fiction
POP Ballads Rom	1,486	482	16	1,325	52	narrative fiction

In the same way, the type of an unknown text can be assessed by comparing the number of sentences to the number of local complements. It remains to be seen whether these local and temporal complements are precise or imprecise.

In the case of dialogue (like in the case of the CHAT texts), the participants relate the information to the communication situation, using the adverbial complements *here* and *now* as well as their synonyms.

Example 2:

Acu doarme aici lângă calculator. (En: *Now she is sleeping here near the computer*.

As the table shows, our social media texts have a big number of temporal complements, but only 103 precise ones, with numerals: *pe 14* En: *On the 14th* (date of the day),*pe la 11*; En: *Around 11* (the hour).

Local modifiers are a bit less numerous, and contain only a small number of geographical names: *la Cairo, la Romexpo, spre Calea Victoriei, de la Iași, în București*. En: *to Cairo, at Romexpo, towards Calea Victoriei, from Iasi, in Bucharest*.

In Orwell's novel "1984", being a fictional narrative, the complements of time and place will be in a small number and without much precision. Only 9 temporal modifiers contain numerals and 10 local modifiers contain proper names. The names are almost all character names. There are also invented geographical names, such as *Airbase One*. By comparison, in FrameNet, where there are articles in journals that report various real facts, the number of time and place modifiers and the number of numerals (38) and proper names (81) is higher. Geographical proper names are present and we can see that they are often territorial subdivisions or precisely delimited confined spaces: *at Roseberry Road, in Ewood Park, to Godstowe, at Greysteel*.

"Flower of Gifts" is a 16th century series of fictional narratives with moral teachings. It has very few local and temporal precise determinations, only 4 temporal modifiers with numerals, and 34 local modifiers with proper nouns. The proper names are of historical leaders or philosophers cited: *Aesop, of Alexander*.

One ontological problem is the classification as a reality narrative of the biblical text. It is historically attested, presented as real by its authors but challenged by some readers. The number of temporal and especially spatial determinations is large, as in reality narratives. The determination by numerals (*Gospel 44, Apostles 59*) and the geographical proper names are frequent and detailed, all the territorial subdivisions covered by Jesus's journey and then the journeys of the Apostles are named. Examples: *In Bethany, in Jerusalem, in Erihon, in Samaria, in Damascus, in Israel, in Mesopotamia, in Cappadocia, in Egypt, in Phrygia* etc. As these statistics show and in accordance with the intention of the emitter's communication, we therefore consider the religious text a real narrative.

"The Chronicle" by Neculce is a reality narrative of the 18th century. This chronicler is a memoirist; he recounts historical events of his life. Being a military leader, he knows precise data about the route taken by various armies and describes them in detail, using chains of local determiners and proper geographical names. The number of local determiners and proper names is higher than in any other text in our database, 2,129 local modifiers with proper names. This specific aspect of the text is related to the memorial character. Being a chronicle, naturally neither of the specified time complements, 144, which contain numerals (days, years), are missing.

As for the popular ballads, these are fictional narratives, the number of time complements is small, and the local modifiers are imprecise, being descriptions of nature that participate in the plot and characterization. Proper names are character names. In the case of other types of text, lyrical, scientific, legal, they will have a small number of spatial and temporal complements, just as in fictional texts, but other classification criteria must also be applied.

7. Conclusions

Time and space verbal modifiers are very important for configuring the textual world. These are the most numerous circumstantial complements in texts, along with the modal ones, which represent a semantically heterogeneous category. In order to prevent

parsers from mistaking time and space verbal modifiers, they need to memorize a large number of examples and the entire training corpus must be correctly annotated. By linking words to a semantic dictionary we can get a better accuracy parser.

In order to construct a classifier of texts, one can count the spatial and temporal complements and one can observe if they represent determinations of exact landmarks (with proper names and numbers) - in which case the text is a real narrative -, or if they represent imprecise determinations, in which case the narrative is fictional. The number and the precision of time and space verbal determiners can demonstrate the real, nonfictional character of a narration.

The classification must be continued adding other criteria, as: the logical relations of condition and consequence in legal texts, the figurative meaning in lyrical texts, the designation of the emitter and receiver in direct speech, etc.

References

Ballesteros, M. and Nivre, J. (2012) MaltOptimizer: An Optimization Tool for MaltParser. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 58–62.

Barbu Mititelu, V., Ion, R., Simionescu, R., Scutelnicu, A. and Irimia, E. (2016). Improving parsing using morpho-syntactic and semantic information. *The Romanian Human-Computer Interaction Journal*, 9(4):285-304.

Bejček, E., Hajičová, E. and Mikulová, M. (2018). The Relation of Form and Function in Linguistic Theory and in a Multi-layer Treebank. In P*roceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, 56–63.

Colhon, M., Mărănduc, C. and Mititelu, C. (2017). A Multiform Balanced Dependency Treebank for Romanian. In *Proceedings of Knowledge Resources for the Socio-Economic Sciences and Humanities, (KnowRSH) workshop at the Recent Advances in Natural Language Processing* (RANLP), 9-18.

Erjavec. T. (2012). MULTEXT-East: Morphosyntactic Resources for Central and Eastern European Languages. *Language Resources and Evaluation*, Vol. 46/1. Netherlands. Springer Publisher, 131–142.

Forăscu, C. and Tufiș, D. (2012) Romanian TimeBank: An Annotated Parallel Corpus for Temporal Information. In *Proceedings of Language Resources and Evaluation,* LREC, 3762-3766.

Hall, J., Nivre, J. and Nilsson, J. (2006): MaltParser: A Data-Driven Parser-Generator for Dependency Parsing. In *Proceedings of the Fifth Intelligence Conference on Language Resources and Evaluation*, LREC, Genoa, Italy, 2216-2219.

Jeong, Y. and Myaeng, S-H., (2012). Using Syntactic Dependencies and WordNet Classes for Noun Event Recognition. In *Proceedings of the premier international forum, for the Semantic Web Linked Data Community*, 41-50.

Kolomiyets, O., Bethard S. and Moens, M.-F. (2012). Extracting Narrative Timelines as Temporal Dependency Structures. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics* (ACL 2012), 88-97.

Lefeuvre-Halftermeyer, A., Antoine, J.-Y., Couillault, A, Schang, E., Abouda, L., Savary, A., Maurel, D., Eshkol-Taravella, I. and Battistelli, D. (2016). Covering various Needs in Temporal Annotation: a Proposal of Extension of ISO TimeML that Preserves Upward Compatibility. In *Proceedings of Language Resources and Evaluation Conference*, 2016, 3802-3806.

Llorens H., Navarro-Colorado, B. and Saquete Boró, E. (2009). From Semantic Roles to Temporal Information Representation. In *Advances in Artificial Intelligence and Soft Computing 14th Mexican International Conference on Artificial Intelligence* (MICAI-2009), 36–43.

Llorens H., Saquete Boró, E. and Navarro-Colorado, B. (2012). Automatic system for identifying and categorizing temporal relations in natural language. *International Journal of Intelligent Systems* 27(7), 680-703.

Mărănduc, C., (2005). *Norms of correct text formation - from a pragmatic perspective*, National Foundation for Science and Art Publishing House, Bucharest.

Mărănduc, C. and Bobicev, V. (2017). Non-Standard Treebank Romania – Republic of Moldova in the Universal Dependencies. In *Proceedings of Conference on Mathematical Foundations of Informatics* (MFOI), 111-116.

Mărănduc, C., Mititelu, C. and Bobicev, V. (2018). Syntactic Semantic Correspondence in Dependency Grammar. In Jan Hajic (Ed.), *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*, 167-180.

Mel'Čuk, I. A., (1987). *Dependency Syntax: Theory and Practice*, Buffalo, Suny Press.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R. and Zeman, D. (2016). Universal Dependencies v1: A Multilingual Treebank Collection. *In Proceedings of Language Resources and Evaluation*, LREC'16, 1659-1666.

Perez, C.-A. (2014). Linguistic Resources for Natural Language Processing. PhD dissertation, Iași, Al. I. Cuza University.

Simionescu, R. (2011). Hybrid POS Tagger. In *Proceedings of the Workshop on Language Resources and Tools in Industrial Applications*, Eurolan 2011 summer school, Cluj-Napoca, Romania, 21-28.

Smith, A., Bohnet, B., Lhoneux, M., Nivre, J., Shao, Y. and Stymne, S. (2018). 82 treebanks, 34 models: Universal dependency parsing with multi-treebank models. In *Proceedings of the CoNLL2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, 113–123.

Strötgen, J. (2010). Extraction and exploration of spatiotemporal information in documents. In *Proceedings of the 6th Workshop on Geographic Information Retrieval*.

Tesnière, L.: (1959) *Elements of structural syntax*, Paris, Klincksieck.

Xiaoping, Ye., Peng, Z. and Guo, H. (2011). Spatio-Temporal Data Model and Spatio-Temporal Databases. *Temporal Information Processing Technology and Its Application*, Springer, Berlin, Heidelberg, 91–112.

ROMANIAN RESOURCES IN LINGUISTIC LINKED OPEN DATA FORMAT

VERGINICA BARBU MITITELU, ELENA IRIMIA, VASILE PĂIȘ, ANDREI-MARIUS AVRAM, MARIA MITROFAN AND ERIC CUREA

Romanian Academy Research Institute for Artificial Intelligence,
{vergi,elena,vasile,andrei.avram,maria,eric}@racai.ro

Abstract

In this paper we present the first steps taken by our team in the process of turning language resources into the linked data format, which ensures their interoperability with other resources in the same standardized format, the possibility of being queried in a similar way and of guaranteeing their consistency. The project is at its debut, so the conversion of only a few language resources is presented herein: the Romanian wordnet, the Romanian Reference Treebank and some corpus-driven linguistic data. These Romanian linguistic linked data resources are available on a dedicated website, while some of the convertors are available on GitHub.

Key words — corpus frequencies, linguistic linked open data, treebank, Romanian, word embeddings, wordnet.

1. Introduction

Human-computer interaction is not a recent scientific development, but it is already about 60 years old, Weizenbaum's Eliza (1966) being its first spectacular embodiment, though rudimentary today. This type of communication, machine-mediated, has widely spread within communities and it is facilitated by language processing tools able to perform satisfactorily within a multimodal environment, trained (and tuned) on linguistic resources. Nowadays, the landscape of such resources is characterised by the rich diversity of the formats in which they are available, by dispersion in location and by difficulty in retrieving.

At the European level there are several similar initiatives which have as goal the development of: infrastructures for language resources and technologies (CLARIN[1]), a network of points for storing language resources documented with quality metadata (ELRC-SHARE[2], META-SHARE[3]), a mechanism for monitoring the new or existent resources or tools (LRE Map[4] created within the FLaReNet[5] project), a scalable cloud platform offering access to linguistic data and technologies, opened to both non-commercial and commercial parties (European Language Grid[6]), an "ecosystem of

[1] https://www.clarin.eu/
[2] https://www.elrc-share.eu/
[3] http://www.meta-share.org/
[4] http://lremap.elra.info/
[5] http://www.flarenet.eu
[6] https://www.european-language-grid.eu/

multilingual and semantically interoperable linguistic data" (Nexus Linguarum COST Action[7]).

In this paper we present the first steps taken by our team in the process of turning language resources into the linked open data (LOD) format. The next section contains a brief presentation of the linked data concept and its principles; section 3 is dedicated to the resources that have already been turned into this format, while in the last section we offer information about access to these resources and conclude the paper, also envisaging some further work.

2. Linguistic Linked Open Data

2.1. Linked Data

The *linked data* concept refers to a set of best practices in publishing structured data on the Web. In order to successfully exploit data, they must be interoperable and integrated with each other (Chiarcos *et al.*, 2013). Ide and Pustejovsky (2010) distinguished between syntactic and semantic interoperability of computer systems, the former relying on "specified data formats, communication protocols, and the like to ensure communication and data exchange", while the latter implies two (or more) systems' ability "to automatically interpret exchanged information meaningfully and accurately". They further adapt the definitions of these two types of interoperability to language resources: syntactic interoperability is "the ability of different systems to process (read) exchanged data either directly or via trivial conversion" and semantic interoperability is "the ability of systems to interpret exchanged linguistic information in meaningful and consistent ways with reference to a common set of reference categories". There are four levels at which interoperability is required: metadata (the same set of metadata fields to which identically defined metadata categories correspond), data categories (they must be standardized, have a unique identifier and a linguistic description), publication of resources (in a standardized format) and software sharing (using the same file format, the same protocols) (Ide and Pustejovsky, 2010).

As far as integration is concerned, data published on the web must stick to a set of four principles (called Linked Data Principles and proposed by Tim Berners-Lee[8]):

(i) using URIs (Uniform Resource Identifiers) as names for things;

(ii) using HTTP URIs so that users can search for those names;

(iii) when searching for URIs, information using Web standards must be provided;

(iv) including links to other URIs so as to reach more things.

2.2. Linguistic Linked Open Data (LLOD)

When applying the above mentioned principles to linguistic data, we talk about *linguistic linked data* and Chiarcos *et al.* (2013) explained the application mechanisms:

[7] https://www.cost.eu/actions/CA18209/#tabs|Name:overview
[8] https://www.w3.org/DesignIssues/LinkedData.html

(i) assign a URI to each element of a resource, thus ensuring the uniqueness of the resource and the possibility of uniquely identifying it;

(ii) the user interested in a resource can obtain (machine- or human-readable) information about it by merely accessing the established protocol (HTTP);

(iii) language resource must be represented in a standardized format (Resource Description Framework, RDF (Klyne *et al*., 2004)) and must be queryable by also using a standardized query language (SPARQL (Prud'hommeaux and Seaborne, 2008));

(iv) links from one resource to another/others will create networks of language(s) resources.

All these principles ensure that language resources are represented as directed labelled graphs and are structurally interoperable. The common format makes different data, with different content, different origin, etc. accessible by means of the same query language. Tools created in one domain can cross it and become usable with resources from other domains. Axioms can be formulated over the vocabulary, thus ensuring annotation consistency (Chiarcos *et al.*, 2013).

When a resource is available with an open type of license, namely Creative Commons (CC), then we talk about *linguistic linked open data* (LLOD).

2.3. The LLOD Cloud

The Linguistic Linked Open Data cloud[9] (LLOD cloud) (McCrae *et al.*, 2016) reflects the extent to which the concept of linked data was adopted by the linguistic community. It contains language resources, as well as linguistically relevant resources. With respect to their types, we find corpora, lexical-conceptual resources (lexicons and dictionaries; terminologies, thesauri and knowledge bases) and metadata (linguistic resource metadata; linguistic data categories; typological databases).

The Romanian language is already present in the LLOD cloud, mainly due to multilingual resources the cloud contains. At the moment of this writing, these are:

- terminological databases (ex.: EuroTermBank),
- parallel (ex.: Europarl) or comparable (ex.: ACCURAT corpus of comparable sentences) corpora,
- lexicons (ex.: Bilingual term pairs extracted from comparable Web resources using the TaaS Bilingual Term Extraction System),
- dictionaries (ex.: Spanish-Romanian LMF Apertium Dictionary).

There are only two monolingual resources:

- CoDII-NPI.ro (a lexicon of Romanian negative polarity items) and
- Romanian Business Corpus (a journalistic corpus containing 21.4 million words, tokenized, morphologically annotated, lemmatized and syntactically annotated using the Constraint Grammar formalism).

[9] https://linguistic-lod.org/llod-cloud

A part of these Romanian resources already existing in the LLOD cloud were created in RACAI (ex.: Strongly Comparable and Aligned Legal News EN-FR-RO News Corpus[10]) or with RACAI's collaboration (ex.: ACCURAT test corpus for renewable energy domain[11], among others).

3. Romanian language resources in LLOD format

We have recently converted some resources created at RACAI to the linked data format, as described below and they are made available on our website.

3.1. The Romanian wordnet

A wordnet is a valuable lexical resource that reflects the lexical richness and expressivity of the language in question and is also extensively used in various NLP tasks. The development of the Romanian wordnet (RoWN) started in the BalkaNet project (Tufiș *et al.*, 2004) with two teams from RACAI, Bucharest and UAIC, Iași and afterwards continued at RACAI. The chosen RoWN development approach was the *expand* method (Vossen, 2002) and it meant importing the structure of the Princeton WordNet (PWN) (Miller, 1995; Fellbaum, 1998) and populating the Romanian synsets with the translation equivalents of the English literals. As RoWN was aligned to PWN, its synsets inherited the identification number (ID) from their corresponding PWN synsets. Implemented based on the Hierarchy Preservation Principle (Tufiș and Cristea, 2002) and the Conceptual Density Principle (Tufiș *et al*., 2004), RoWN currently numbers 56,591 lexicalised synsets, comprising 53,092 literals.

It was created and maintained in an XML format, with a DTD (Document Type Definition) designed according to the BalkaNet principles[12]. To convert this resource to an LLD-compatible format, we followed the instructions and schemas of the Best Practices for Multilingual Linked Open Data Community Group[13]. The Lexicon Model for Ontologies (LEMON), dedicated to representing lexicons relative to ontologies, is the recommended standard for wordnets and facilitates their linking to Semantic Web or LLOD Cloud. The main LEMON elements adopted to represent RoWN are the *lexical entry* (mono- or multi-word), the *lexical sense* (represents one of the meanings of the lexical entry and contains a reference to a *synset* in the network). The SKOS vocabulary and a wordnet-specialised vocabulary[14] are used to introduce a Synset type as a subclass of SKOS Concept class and to express synset relations. Prior to converting the resource into the three recommended LLOD formats (XML/LMF, JSON and Turtle RDF), some processing was necessary to: 1. map RoWN to ILI[15] (through the PWN mapping); 2. import subcategorization frames from RoWN2.0 to RoWN3.0; 3. rename lexical and

[10] http://linghub.org/metashare/c8a540a0fb6711e2a8ad00237df3e3584159e2a550584f6d9af53132b5aeebeb#contactPerson
[11] http://linghub.org/metashare/78358384a37611e3960f001dd8b71c195469cc63784a411c95871d3bd8b58b20
[12] http://www.dblab.upatras.gr/balkanet/deliverables/d.2.pdf
[13] http://bpmlod.github.io/report/WordNets/index.html
[14] http://wordnet-rdf.princeton.edu/ontology#
[15] The collaborative interlingual index of concept for wordnets https://raw.githubusercontent.com/globalwordnet/ili/master/ili-map-pwn31.tab

semantic relations to correspond to the LLOD guidelines. 58,807 of 59,348 RoWN synsets were mapped to the ILI, while for the 541 Balkan specific concepts, which do not have an ILI correspondent, the *ili* attribute in the document remained empty (the corresponding lexical entries are accessible through lexical search).

The *expand* method comes with a collateral issue: lexical relations are difficult to transfer from one language to another, being set between lexicalisation of concepts (which are specific to language) and not between concepts (or synsets). There are more relations of this type (*antonym, also_see, derivat, derived_from, pertainym, participle*) in the RoWN, but for this step of LLOD integration we dealt only with *antonym*, that we transferred at synset level, based on the fact that it is a lexical relation that reflects also a conceptual opposition. The other lexical relations are to be imported in LLOD RoWN in a future stage: *e.g.*, derivational relations that were marked as stand-off annotations (Barbu Mititelu, 2013).

The actual version of LLOD RoWN (3.0), in the three formats that we mentioned (XML/LMF, JSON and Turtle RDF[16]), is available for download on the project's site. An example of an entry in each of these formats is given in the Appendix to this paper.

3.2. The Romanian Reference Treebank

The Romanian Reference Treebank (RoRefTrees or RRT) (Mititelu *et al.*, 2016) contains sentences taken over from two previous treebanks: UAIC-RoDepTb (Perez, 2014) and RACAI-RoTb (Irimia and Barbu Mititelu, 2015). It comes in the CoNLL-U format[17], a revised version of the CoNLL-X format (Buchholz and Marsi, 2006), and is organised in three splits: the training set (75%), the development set (12.5%) and the test set (12.5%). RoRefTrees contains 9,523 sentences and 218,511 tokens, extracted from texts in 9 genres, distributed in an unbalanced manner: 19.09% literature, 16.86% law, 12.70% medical, 11.46% FrameNet translations, 9.97% academic writing, 9.79% news, 3.80% science, 2.63% wikipedia and the rest from miscellaneous sources. The dataset comes under the CCA 4.0 license, and is publicly available[18].

The RoRefTrees dataset was also converted into the format specified by LLOD, namely RDF/XML, by using the CoNLL-U to RDF tool proposed by the Applied Computational Linguistics Lab (ACoLi), Goethe University Frankfurt (Chiarcos and Fäth, 2017). However, the tool converted the file in a Turtle format, so we further used a Python script[19] to convert[20] it to XML/RDF. We preferred this conversion variant from Turtle to XML/RDF because all the standard conversion tools that we experimented with randomized the sentences or/and the tokens inside a sentence, which is to be avoided as the sentences in the treebank are grouped according to their genre. An example of the beginning of a sentence, a token and the end of a sentence from RoRefTrees in XML/RDF format are offered in Figure 1.

[16] following schemas from https://github.com/globalwordnet/schemas
[17] https://universaldependencies.org/format.html
[18] https://github.com/UniversalDependencies/UD_Romanian-RRT
[19] https://github.com/racai-ai/RoLLOD/tree/master/conllu_convertors
[20] The conversion scripts are available at https://github.com/racai-ai/RoLLOD

```
<rdf:Description rdf:about="https://github.com/UniversalDependencies/UD_Romanian-RRT/tree/r2.7#rrt_train_s1_0">
        <rdf:type rdf:resource="http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core#Sentence"/>
</rdf:Description>
```

```
<rdf:Description rdf:about="https://github.com/UniversalDependencies/UD_Romanian-RRT/tree/r2.7#rrt_train_s1_1">
        <rdf:type rdf:resource="http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core#Word"/>
        <conll:WORD>Lui</conll:WORD>
        <conll:EDGE>det</conll:EDGE>
        <conll:FEAT>Case=Dat,Gen|Definite=Def|Number=Sing|PronType=Art</conll:FEAT>
        <conll:HEAD rdf:resource="https://github.com/UniversalDependencies/UD_Romanian-RRT/tree/r2.7#rrt_train_s1_2"/>
        <conll:ID>1</conll:ID>
        <conll:LEMMA>lui</conll:LEMMA>
        <conll:POS>Tf-so</conll:POS>
        <conll:UPOS>DET</conll:UPOS>
        <nif:nextWord rdf:resource="https://github.com/UniversalDependencies/UD_Romanian-RRT/tree/r2.7#rrt_train_s1_2"/>
</rdf:Description>
```

```
<rdf:Description rdf:about="https://github.com/UniversalDependencies/UD_Romanian-RRT/tree/r2.7#rrt_train_s1_0">
        <nif:nextSentence rdf:resource="https://github.com/UniversalDependencies/UD_Romanian-RRT/tree/r2.7#rrt_train_s2_0"/>
</rdf:Description>
```

Figure 1: The beginning of a sentence (up), a token (middle) and the end of a sentence (down) from RoRefTrees in XML/RDF

3.3. Corpus-driven linguistic data

Ontology Lexica community group (OntoLex) has developed a module for frequency, attestation and corpus information (OntoLex-FRAC) of LEMON. The module is targeted at complementing dictionaries and other linguistic resources containing lexicographic data with a vocabulary to express corpus-derived statistics (frequency and co-occurrence information, collocations), pointers from lexical resources to corpora and other collections of text (attestations), the annotation of corpora and other language resources with lexical information (lemmatization against a dictionary), and distributional semantics (collocation vectors, word embeddings, sense embeddings, concept embeddings). Complete specifications and discussions are available on GitHub[21].

For the purposes of this work, we analysed the available resources associated with the Corpus of Contemporary Romanian Language (CoRoLa) (Barbu Mititelu *et al.*, 2019) and tried to identify those that can be transformed using the OntoLex-FRAC specifications. First, we considered frequency lists (word frequency and lemma frequency), which were initially available as comma separated values (CSV) files. Then, we considered word representations, in the form of word embeddings. These were initially constructed using the algorithms of Bojanowski *et al.* (2017) as implemented in the FastText[22] tool.

Multiple word embeddings representations, trained with different parameters, are available for the CoRoLa corpus and they can be freely downloaded[23] and interrogated (Păiș and Tufiș, 2018b). However, for conversion to OntoLex-FRAC specifications we considered only the best performing model as described by Păiș and Tufiș (2018a). This model considers a vector size of 300 floating point numbers and includes only words appearing in the corpus at least 20 times. Thus, the model offers representations for

[21] https://github.com/ontolex/frequency-attestation-corpus-information
[22] https://fasttext.cc/
[23] http://corolaws.racai.ro/word_embeddings/

250,942 words. Additional representations can be computed by using subword information (Bojanowski *et al.*, 2017).

In order to transform existing resources into OntoLex-FRAC compliant representations, we developed two converters. The first one takes a frequency CSV file and produces LEMON format using the *frac:CorpusFrequency* annotations. The second converter accepts the word embedding representation as text vectors and converts it into LEMON format using the *frac:Embedding* class.

From a technical point of view, the converters were written in Java and, in order to reduce the space required, accept conversion from both raw text files and from compressed, gzipped, text files. For the output, we considered the RDF Turtle[24] format with OntoLex-FRAC components. Similar to the input, the output file can be produced either as raw RDF or compressed RDF (using gzip compression) in order to reduce the space required for storage.

A sample from a converted frequency file in LEMON RDF format is presented in Ex. 1. According to the specifications, it starts with declaring the namespace prefixes used throughout the file, followed by an entry for each word. Each entry contains a "*frac:CorpusFrequency*" element providing the actual number of occurrences in the corpus: from the given example we find out that the word "este" has 3,369,971 occurrences in CoRoLa.

Example 1:

```
@prefix :    <http://corola.racai.ro/> .
@prefix dct:   <http://purl.org/dc/terms/>.
@prefix frac:  <http://www.w3.org/ns/lemon/frac#> .
@prefix ontolex: <http://www.w3.org/ns/lemon/ontolex#> .
@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd:   <http://www.w3.org/2001/XMLSchema#> .

:este a ontolex:LexicalEntry;
 ontolex:canonicalForm "este"@ro;
 frac:frequency [
  a frac:CorpusFrequency;
    rdf:value "3369971"^^xsd:int;
  dct:source <http://corola.racai.ro/> ].
```

A sample from the resulting word embeddings file in LEMON RDF format is presented in Example 2. Given the size of the embeddings vector (300), the list of values was truncated for the purposes of this example. Additionally, a single word, the preposition "de", was included. Similar to the previous example, it starts with declaring the prefixes, as indicated by the RDF specification, followed by declaration of the

[24] https://www.w3.org/TR/turtle/

embedding representation and finally an entry for each word included in the representation. Inside each entry, the "*frac:embedding*" element contains the actual vector representation using float values separated by spaces.

Example 2:

```
@prefix :    <http://corolaws.racai.ro/word_embeddings/> .
@prefix dct:   <http://purl.org/dc/terms/>.
@prefix frac:  <http://www.w3.org/ns/lemon/frac#> .
@prefix ontolex: <http://www.w3.org/ns/lemon/ontolex#> .
@prefix owl:   <http://www.w3.org/2002/07/owl#> .
@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd:   <http://www.w3.org/2001/XMLSchema#> .

:CoRoLaEmbeddings_300 rdfs:subClassOf frac:Embedding;
 rdfs:subClassOf
  [ a owl:Restriction;
     owl:onProperty dct:source;
     owl:hasValue
        <http://corolaws.racai.ro/word_embeddings/> ],
  [ a owl:Restriction;
     owl:onProperty dct:extent;
     owl:hasValue "300"^^xsd:int ],
    [ a owl:Restriction;
     owl:onProperty dct:description;
     owl:hasValue "Word Embeddings from the CoRoLa corpus"@en ].

:de a ontolex:LexicalEntry;
 ontolex:canonicalForm "de"@ro;
 frac:embedding [
  a :CoRoLaEmbeddings_300;
    rdf:value "0.058826 0.050749 0.094646 0.059437 0.014913 -0.14699 0.31223
0.25699 0.020498 0.1497 -0.045657 -0.16574 -0.14085 0.053746 -0.016113 -
0.11879 0.11086 -0.086826 -0.11564 -0.1137 -0.21041 0.12873 0.074748 -0.1439 -
0.11781 -0.14723 0.080661 0.18918 0.079647 -0.043609 -0.024831 0.058612
0.0028617 0.074098 0.048036 .....
```

Converter implementation is available as open source[25] and the resulting LEMON RDF resources are available for download on the RoLLOD website (see the next section).

[25] https://github.com/racai-ai/RoLLOD

Due to the nature of the RDF format, such as the inclusion of a heading and the repetition of *rdf:value* tags, the overall uncompressed size of the word embedding representation has increased from 662,153,768 bytes (631Mb) to 694,556,958 bytes (662Mb). This increase accounts for 31Mb (4% of the output file).

4. Conclusions

Tools and applications making use of linguistically annotated data face the problem of lack of interoperability: different resources use different annotation schemes and different vocabularies. This holds true when considering resources for different languages, but also when considering resources for the same language and leads to slow progress in their development. Linked data is meant to overcome this shortcoming by reusing the same vocabularies and interpreting them against a common basis. We have presented here the first steps in creating convertors for our resources or using existing ones so as to make a first batch of resources available in LLOD format: the Romanian wordnet, the Romanian Reference Treebank and some corpus-driven linguistic data. Converters for corpus-driven linguistic data are designed to be general purpose ones, accepting different configuration options apart from the input files. Thus, they can be used in other projects as well for converting similar data into LEMON RDF format. The resources converted to the LLOD format are made available on a dedicated page on the RACAI's website: http://www.racai.ro/p/llod/index.html, whereas the conversion tools are available in a GitHub repository: https://github.com/racai-ai/RoLLOD.

Apendix. Example of one of the senses of the word *conferință* as an entry in all three LLOD formats of the RoWN:

1. XML/LMF

```
<LexicalEntry id="rown-conferință-n">
   <Lemma writtenForm="conferință" partOfSpeech="n"/>
   <Senseid="rown-conferință-n-07142566-1.2.x" synset="rown-07142566-
n"></Sense>
</LexicalEntry>
<Synset id="rown-07142566-n" ili="i74199" partOfSpeech="n">
<Definition>Reuniune a reprezentanților unor state, ai unor organizații politice, 
științifice etc., cu scopul de a dezbate și de a hotărî asupra unor probleme curente și de 
perspectivă ale activității lor</Definition>
   <SynsetRelation relType="hypernym" target="rown-07140659-n"/>
   <SynsetRelation relType="hyponym" target="rown-07143137-n"/>
                  <SynsetRelation relType="hyponym" target="rown-07143624-n"/>
                  <SynsetRelation relType="hyponym" target="rown-07144416-n"/>
                  <SynsetRelation relType="hyponym" target="rown-07144834-n"/>
                  <SynsetRelation relType="hyponym" target="rown-07145314-n"/>
                  <SynsetRelation relType="hyponym" target="rown-07145508-n"/>
                  <SynsetRelation relType="hyponym" target="rown-07145783-n"/>
                  <SynsetRelation relType="hyponym" target="rown-07145958-n"/>
</Synset>
```

2. JSON

{ "@id" :"rown-conferință-n",
"lemma": { "writtenForm": "conferință" },
"partOfSpeech":"n",
"sense":
[{ "@id": "rown-conferință-n-07142566-1.2.x",
"synsetRef": "rown-07142566-n"
}]},
{ "@id":"rown-07142566-n",
"ili": "i74199",
"partOfSpeech":"n",
"definition": [{"gloss": "Reuniune a reprezentanților unor state, ai unor organizații politice, științifice etc., cu scopul de a dezbate și de a hotărî asupra unor probleme curente și de perspectivă ale activității lor"}],
"relations":[
{"relType": "hypernym", "target": "rown-07140659-n" },
{"relType": "hyponym", "target": "rown-07143137-n" },
{"relType": "hyponym", "target": "rown-07143624-n" },
{"relType": "hyponym", "target": "rown-07144416-n" },
{"relType": "hyponym", "target": "rown-07144834-n" },
{"relType": "hyponym", "target": "rown-07145314-n" },
{"relType": "hyponym", "target": "rown-07145508-n" },
{"relType": "hyponym", "target": "rown-07145783-n" },
{"relType": "hyponym", "target": "rown-07145958-n" }]},

3. TurtleRDF

<#rown-conferință-n> a ontolex:LexicalEntry ;
ontolex:canonicalForm [
ontolex:writtenRep "conferință"@ro] ;
wn:partOfSpeech wn:n ;
ontolex:Sense <#rown-conferință-n-07142566-1.2.x> .
<#rown-07142566-n> a ontolex:LexicalConcept ;
wn:partOfSpeech wn:n ;
skos:inScheme <#rown> ;
wn:ili ili:i74199 ;
wn:definition [rdf:value "Reuniune a reprezentanților unor state, ai unor organizații politice, științifice etc., cu scopul de a dezbate și de a hotărî asupra unor probleme curente și de perspectivă ale activității lor"@ro] .
[]
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hypernym ;
vartrans:target <#rown-07140659-n> ;
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hyponym ;
vartrans:target <#rown-07143137-n> ;
vartrans:source <#rown-07142566-n> ;

vartrans:category wn:hyponym ;
vartrans:target <#rown-07143624-n> ;
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hyponym ;
vartrans:target <#rown-07144416-n> ;
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hyponym ;
vartrans:target <#rown-07144834-n> ;
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hyponym ;
vartrans:target <#rown-07145314-n> ;
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hyponym ;
vartrans:target <#rown-07145508-n> ;
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hyponym ;
vartrans:target <#rown-07145783-n> ;
vartrans:source <#rown-07142566-n> ;
vartrans:category wn:hyponym ;
vartrans:target <#rown-07145958-n> .

References

Barbu Mititelu, V. (2013). *Rețea semantico-derivațională pentru limba română. Bucharest*, Muzeul Literaturii Române Publishing House.

Barbu Mititelu, V., Ion, R., Simionescu, R., Irimia, E. and Perez, C.A. (2016). The Romanian Treebank Annotated According to Universal Dependencies. In *Proceedings of the tenth international conference on natural language processing*.

Barbu Mititelu, V., Tufiș, D., Irimia, E., Paiș, V., Ion, R., Diewald, N., Mitrofan, M., Onofrei, M. (2019). Little Strokes Fell Great Oaks. Creating CoRoLa, The Reference Corpus of Contemporary Romanian. *Revue roumaine de linguistique*, LXIV, 3, 227-240.

Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135-146.

Buchholz, S. and Marsi, E. (2006). CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the tenth conference on computational natural language learning*, 149-164.

Chiarcos, C., McCrae, J., Cimiano, P. and Fellbaum, C. (2013). Towards open data for linguistics: Linguistic linked data. In *New Trends of Research in Ontologies and Lexical Resources*, Springer, Berlin, Heidelberg, 7-25.

Chiarcos C. and Fäth C. (2017). CoNLL-RDF: Linked Corpora Done in an NLP-Friendly Way. In Gracia, J., Bond, F., McCrae, J., Buitelaar, P., Chiarcos, C.,

Hellmann, S. (eds), *Proceedings of Language, Data, and Knowledge*, Springer, LNAI, 74-88.

Fellbaum, Ch. (ed.) (1998). *WordNet: an electronic lexical database*. Cambrige, MIT Press.

Ide, N. and Pustejovsky, J. (2010). What does interoperability mean, anyway? Toward an operational definition of interoperability for language technology. In *Proceedings of the Second International Conference on Global Interoperability for Language Resources.* Hong Kong, China.

Irimia, E., Barbu Mititelu, V. (2015). RACAI-RoTb: nucleu de corpus de limbă română adnotat sintactic cu relaţii de dependenţă, *Revista Română de Interacţiune Om-Calculator* 8 (2) 2015, 101-120.

Klyne, G., Jeremy, J.C., and McBride, B. (2004). Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation, Feb.

McCrae, J.P., Chiarcos, C., Bond, F., Cimiano, P., Declerck, T., De Melo, G., Gracia, J., Hellmann, S., Klimek, B., Moran, S. and Osenova, P. (2016). The open linguistics working group: developing the Linguistic Linked Open Data cloud. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2435-2441.

Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM*, 38, 11, 39-41.

Păiș, V., Tufiș, D. (2018a). Computing distributed representations of words using the CoRoLa corpus. *Proceedings of the Romanian Academy, series A*, 19, 2, 403-409.

Păiș, V. and Tufiș, Dan. (2018b). More Romanian word embeddings from the ReTeRom project. In *Proceedings of the International Conference on Linguistic Resources and Tools for Processing Romanian Language - CONSILR*, 91-100.

Perez, C.A. (2014). *Linguistic Resources for Natural Language Processing*, PhD dissertation, A.I. Cuza University of Iasi (in Romanian).

Prud'hommeaux, E. and Seaborne, A. (2008). SPARQL Query Language for RDF. W3C working draft, 15 January 2008.

Tufiș, D. and Cristea, D. (2002). Methodological Issues in Building the Romanian Wordnet and Consistency Checks in BalkaNet. In *Proceedings of LREC2002 Workshop on Wordnet Structures and Standardisation*, 35-41.

Tufiș, D., Cristea, D., Stamou, S. (2004). BalkaNet: Aims, Methods, Results and Perspectives. A General Overview. *RomJIST*, 7 (1-2), 9-43.

Vossen, P. (ed.). (2002). EuroWordNet General Document, version 3.

Weizenbaum, J. (1966). ELIZA - A Computer Program For the Study of Natural Language Communication Between Man And Machine. *Computational Linguistics*, 9, 1, 36-45.

THE LECOR PROJECT. A PRESENTATION

CARMEN MÎRZEA VASILE

University of Bucharest, Faculty of Letters, Department of Linguistics

carmen.vasile@unibuc.ro

Abstract

The project *Learner Corpus of Romanian (LECOR). Collection, Annotation and Applications* (PN-III-P1-1.1-TE-2019-1066) is grant-aided by the Executive Unit for Financing Higher Education, Research, Development and Innovation (UEFISCDI), as part of the subprogramme dedicated to research projects to stimulate young independent teams (TE). The paper presents the characteristics of LECOR and the specific objectives of the project, as well as the research methodology. Two topics are tackled in more detail: the ethical issues that arise from the activities carried out (the collection of a learner corpus, its storage, data processing and post-processing, etc.) and the LECOR metadata. The linguistic annotation that will be implemented, the error taxonomy and the specific approach to error annotation will not be presented in more detail, because the team members are still at the documentation stage and they have not made final decisions on all these issues.

Key words — building a learner corpus, learner corpus for Romanian, learner metadata, Romanian as a Foreign/Second Language, text and task metadata

1. Introduction

The project *Learner Corpus of Romanian (LECOR). Collection, Annotation and Applications* will be carried out over a period of two years starting from 15th September 2020, at the University of Bucharest (The Solomon Marcus Centre for Computational Linguistics[1], of the Faculty of Letters). The project team has seven members (2 IT specialists and 5 linguists, all five of whom teach RFL (Romanian as Foreign Language) at the University of Bucharest, in the Preparatory Year intensive programme dedicated to foreign students).

A learner corpus is an electronic collection of authentic FL (Foreign Language)/SL (Second Language) textual data, created according to explicit design criteria for particular SLA (Second Language Acquisition)/FLT (Foreign Language Teaching) purposes. It is encoded in a standardised and homogeneous way and is documented as to its origin and provenance (Granger, 2002). There has been an increasing interest in digitalized learner corpora since the late eighties. There are now over 150 learner corpora all around the world[2]; most of them are for English (*Cambridge Learner Corpus, International Corpus of Learner English, The Barcelona English Language Corpus*), but corpora have also been compiled for French (*French Learner Language Oral Corpora, Corpus Écrit de Français Langue Étrangère*), German (*Fehlerannotiertes Lernerkorpus, What's Hard in German?*), Spanish (*Corpus Escrito*

[1] http://clc.litere.ro

[2] See the list at https://www.uclouvain.be/en-cecl-lcworld.html

del Español L2), Finnish (*International Corpus of Learner Finnish*), etc. There is also a very rich literature on the subject of learner corpora; researchers have discussed the corpus, its design, collection, digitalisation, applications etc. (for some overviews, see Pravec, 2002; Granger, 2002; Rastelli, 2009; Díaz-Negrillo and Thompson, 2013; Granger *et. al.*, 2015).

Traditionally, learner corpora are used for didactic purposes: to design course books (Brook-Hart, 2009), learner dictionaries (Rundell, 2007), wordlists (Schmitt and Schmitt, 2011) etc. No such materials exist for Romanian, a language that is hard to learn due to its rich morphology. Such didactic tools have been requested by foreign students over the past years, therefore the learner corpus is a both useful and urgent endeavour.

There is no available digitalised learner corpus for Romanian, though, in the past, isolated attempts have been made. However, none of these corpora have been annotated, and, furthermore, no such collections of Romanian interlanguage have been of a significant size up to now. Recently, some authors (Bocoș, 2017; Arieșan and Vasiu, 2016; Gafu *et. al.*, 2012) have mentioned using small learner corpora in their studies, but these authors do not provide a detailed description of these corpora which were, apparently, created in-house. In the ’80s, within the context of the international project RECAP (*The Romanian-English Contrastive Analysis Project*), a learner corpus for Romanian was planned (Constantinescu and Stoica, 2020). There is also an online database of sound samples representative for Romance varieties (RPD – *The Romance Phonetics Database*[3]; see also Constantinescu and Stoica, 2020). Two recent learner corpora in printed format should also be mentioned. The first of these is the CORLS – *Corpus Oral de Limba Română ca Limbă Străină* (Vasiu, 2020). Vasiu (2020) collected spoken samples from 172 A1 learners on proficiency tests from 2014 to 2017 at Babeș-Bolyai University (Cluj-Napoca). The transcriptions summarise 70,000 words (Vasiu, 2020). The second corpus was collected and transcribed by Constantinescu and Stoica and was published by the University of Bucharest Publishing House in the same year, 2020. This corpus (*Româna ca limbă străină. Corpus*) contains 380 written samples (65,000 words) and 79 oral transcriptions (60,000 words), collected from 61 A1-B2 learners over the period 2004-2016.

Romanian is a relatively small language, spoken as mother tongue by more than 25 million people (Maiden, 2016). It is a second language for around another 4 million speakers. Since 2007, Romanian has been one of the 24 official languages of the E.U. Due to its particular grammatical and lexical features, Romanian is one of the most interesting Romance languages. Besides programmes at different cultural institutes, organizations and language schools, Romanian as FL/SL is studied intensively in 25 universities in Romania and in about 51 universities around the world. There is an increasing interest in RFL, both within and outside Romania. Unfortunately, neither current pedagogical resources nor second language research meets the requirements of this trend (Mîrzea Vasile, 2016, 2017; Geană, 2018). A learner corpus of Romanian, meeting the current scientific requirements, and studies based on it, will improve the situation considerably.

[3] http://rpd.chass.utoronto.ca/

While collecting samples of learners' work which contain errors may seem not very challenging to some researchers, developing a digitalised learner corpus of a reasonably large size and which is available in open-access format is certainly no easy task. It is a laborious step-by-step process, that presents difficulties as regards to the quantity of data to be dealt with, ethics, the transcription of handwritten and spoken samples, annotation, the registration of learner/task-variables, etc.

2. The objectives of the LECOR project

The main goals of the project are: 1) to collect a rich, raw, learner corpus of Romanian, complying with relevant legal regulations and fulfilling scientific requirements; 2) to annotate a part of the texts; 3) to develop an interface that makes LECOR available to everybody and that allows diverse queries; 4) to carry out studies in learner corpus, FLT and SLA fields; 5) in conjunction with the above, to improve the project staff's experience, particularly that of the young members (enabling them to acquire in-depth knowledge in the afore-mentioned fields and to become acquainted with the international scientific and academic environment) and to increase the international visibility of Romanian research in this specific domain.

LECOR is designed to have the following characteristics: (a) it is a monolingual corpus; (b) it contains written (80%) and oral (20%) learners' samples; (c) the sample collection is mainly controlled (*i.e.* it consists of dialogues, essays, etc., created in the classroom, as homework or during exams); (d) it can be used both as a synchronic/cross-sectional corpus (it contains samples produced by different learners on the same or on different topics in the same period of time) and as a diachronic/longitudinal corpus (as it contains the same learners' productions throughout the entire academic year); (e) as for its breadth, it is mainly a general corpus, but we intend to collect also some B1-B2 samples of language for specific purposes; (f) it is automatically lemmatized and annotated at the morpho-syntactic and syntactic level and manually annotated by linguist specialists for learner morphological and syntactical errors. The relevant literature (for reasons, see Díaz-Negrillo and Thompson, 2013) recommends that the best and most accurate method of error annotation of a learner corpus is the manual type.

The specific objectives of the project are:

- collect various samples produced by learners of Romanian, enrolled for the preparatory language intensive programme. Over the two years, the four postdoctoral researchers in our team will collect at least 4,000 samples (for the type of the samples, see above (a)-(e)). While collecting, storing, and processing the data for LECOR, we will be fully aware of all possible ethical issues (see details in section 5 below). We will draw up privacy statements to be signed both by learners and the team members.
- carefully register the learners' variables and the variables of the learners' productions (task and language-related), in order to allow fine and reliable studies (see detail in section 4 below).
- digitalize LECOR by: (a) transcription of handwritten samples and of oral productions (more than 4,000 samples, in total); in both cases, special attention will

be paid to sentence segmentation, which is a problematic issue in such type of samples; (b) converting the transcriptions from the original format (Word) to a non-formatted format (.txt) and using the RELATE/TEPROLIN platform (Ion, 2018) to lemmatize and morpho-syntactically and syntactically annotate the corpus. The file format produced by this process will be the CONNL-X/U[4], chosen for its simplicity and accessibility for any non-IT specialist. The corpus could be automatically converted into the TEI/XML format at the end, if this format is required in specific research communities; (c) automatically creating metadata files for each transcript, based on the learners' variables and the variables of the learners' productions in the learners' profiles; (d) defining the morphological and syntactic error taxonomy, based on state-of-the art research and work in the international scientific community (see Lüdeling and Hirschmann, 2015; Van Rooy, 2015) and on the errors observed in the samples already collected by the postdoctoral researchers in the previous years. The taxonomy could be enriched with Romanian specific error types identified in the subsequent error annotation process; (e) manually annotating approx. 3,500 sentences from the 4,000 text samples with the error types identified in the taxonomy and other new types. If necessary, an accessible text editor will be used by linguists to annotate directly in the CONNL-X/U format; (f) making LECOR an open-access resource through a web interface. In fact, the database will have two interfaces: one for uploading data and the other for interrogation of the annotated texts. The interface for uploading will contain fields for entering the relevant metadata relating to the learner and fields for entering the metadata related to the text itself. Besides simple queries, it will be possible to carry out searches using many criteria: a type of error (for example, syntactic error in agreement + learners with a certain native language), or a specific error (plural ending, word order of adjectives, etc.).

- Based on LECOR, we also intend to carry out studies and to produce articles as to the different types of errors made by learners of Romanian.

3. The methodology

In the LECOR project, we will use established, specific methods for learner corpora design, collection, annotation and exploitation (see Chaudron, 2003; Callies, 2015; Tono, 2016; Gilquin, 2015). Since LECOR is an open-access resource, it can be used for queries by anyone interested, through a text retrieval software programme. The annotated part of LECOR will allow a broader range of queries (for a type of error, for a part of speech etc.). We will use LECOR with a corpus-informed, corpus-based and corpus-driven approach. Thus, LECOR will be a general reference source for information (*e.g.* the frequency of a certain word, phrase or of a certain type of error, etc.). It will serve to test research hypotheses and exemplify them, and it will help to provide accurate statistical analysis on interlanguage. LECOR will be explored for analyses of both a quantitative (*i.e.* statistics) and qualitative nature (*i.e.* the broad social and philological context of language learning; connections between variables etc.). LECOR will be used cross-sectionally and longitudinally. For our studies regarding

[4] https://universaldependencies.org/format.html

learners' errors, proficiency level, etc., besides the information from Romanian and international literature, we will combine information from the learner corpus and material derived from specific experimental methods (*i.e.* written questionnaires, tests, etc.) aimed at cross-checking the corpus data.

4. The LECOR metadata

Díaz-Negrillo and Thompson (2013) observe that, in general, learner corpora are not well documented and do not contain enough information about the learners and their productions. The literature on interlanguage corpora offers some information about variables/metadata (for an overview, see Granger and Paquot, 2017). Such information can be obtained also by consulting the interfaces of learner corpora that are available online.

LECOR has been planned from the beginning to ensure that it is a very well documented resource, its variables being consistently controlled in the course of building the corpus. These variables are associated to, on the one hand, (1) the learners (sociolinguistic variables, psychological and temperamental variables, variables related to academic performance) and, on the other hand, (2) the type of text and the circumstances in which it was produced (the topic of the text, its length, whether it has been required as part of homework or in the course of an exam, etc.). The recording of such variables will be completed with full regard to all regulations concerning legal requirements and ethics, both in the course of the research and in the teaching process (see section 4 below). Each learner will receive a code which is anonymized (but which relates to their native language and to their proficiency level, etc.). Similarly, each text will be given a code. There will be a correlation between the learner's metadata and those of the texts.

The variables concerning the learners and the texts and tasks are recorded in the metadata, with which the corpus will be annotated. The majority of metadata will be used as criteria for searching the corpus. Other classes of metadata will just be of an informative nature (this category of metadata cannot be used for searches in the corpus, but will offer possibly useful information for more in-depth analyses). In the case of metadata used to interrogate the corpus, there are two modes of filling in the exact value of the variable about which to make a query. These two categories can be used separately or in conjunction. A choice can be made from a list of variants. It will be possible to select more than one variant from the list. Alternatively, the learner can type the appropriate variant into a blank field.

Regarding the five core metadata for learner corpora proposed by Granger and Paquot (2017) – administrative metadata, corpus design metadata, annotation metadata, text metadata, learner metadata –, we will now set out the most important types of learner and text metadata with which our proposed corpus will be annotated. In order to allow sociolinguist research and also psycho-behavioural studies, LECOR learner and text metadata is more varied than that of other similar corpora; see, for example, the metadata of The Louvain International Database of Spoken English Interlanguage (LINDSEI) – task metadata: genre, duration, three tasks, institution; learner metadata: learning context, proficiency, age, gender, L1, country, other FLs, stay in Enghish-

speaking country, or of The International Corpus of Learner English (ICLE) – task metadata: medium, genre, field, length, topic, task setting; learner metadata: age, gender, mother tongue, region, other LFs, stay in English-speaking country, learning context, proficiency level (Granger and Paquot, 2017).

4.1. Learner metadata

In this category, two types of metadata are considered important and will be detailed below (metadata under B and C below are only relevant in the case of Romanian language studies in an institutional setting).

A. Learner metadata completed by the teacher with the help of the student:

- age;
- gender (with four options: male, female, other (please fill in if you want), and rather not say);
- region where the learner is learning Romanian (because we hope that the LECOR corpus will be enriched after the end of the project with samples from learners that are learning Romanian abroad);
- native language or mother tongue (the country of normal residence, the place/country of birth, and the ethnicity are informative metadata);
- if bilingual (trilingual) students, their other native language(s);
- motivation (for their studies, for business purposes, to obtain citizenship, personal interest, etc.);
- knowledge of other foreign language(s) and proficiency level (beginner, intermediate or advanced);
- whether the learner is studying another foreign language in parallel with Romanian;
- general level of education (are they primary or secondary school students, type of secondary school attended, have they completed studies at bachelor's, master's, or doctoral level, etc.);
- their mode of study of L2 Romanian (in an academic context, at school/university; on a private course; informally, just through the process of immersion; using online platforms and mobile applications, etc.);
- in the case of study at an institution, the number of hours spent in class each week;
- again in an institutional setting, what sort of a course was it (general language or language for specific purposes, *e.g.* medical language, legal language);
- whether, prior to the beginning of the process of collecting samples from him/her, the learner had already studied Romanian; if so, the learner will be asked to contribute some details that would have the status of only informative items of metadata, *e.g.* in what kind of programme had they studied, how many lessons a week, etc.;
- whether the learner speaks Romanian in their family/home context;
- their methods of learning Romanian outside the classroom (listening and watching Romanian TV programmes, movies, etc.; listening to music; talking with native speakers in Romanian; self-instruction; listening to audiobooks; using online platforms, resources accessed on mobiles; private classes with a teacher, etc.);

- how much the learner uses Romanian in their interactions with native speakers (not at all; very little; a little; quite a lot; a lot; most of the time; the whole time).

B. Learner metadata to be completed by their teacher and which refers to their academic performance and other aspects of their performance as a student:

- general language aptitude/ abilities (with the variants: 10 – excellent, 9 – very good, 8 – good, 6-7 – satisfactory, 5 – sufficient, 1-4 – unsatisfactory);
- the regularity of attendance at class (with the variants: 100% – excellent, 90% – very good, 80% – good, 60-70% – satisfactory, 50% – sufficient, 10-40% – unsatisfactory).

C. Learner metadata to be completed by their teacher and which refers to the learner's personality and temperament or motivation (as to the utility of this kind of controlled variables, see (Dörnyei, 2005)): the teachers (project members) were asked to give a score from 0-5 for the following learner characteristics: conscientiousness, creativity, extroversion (introversion) and motivation.

In addition, teachers will record descriptive metadata such as: general remarks about the learner's group (the general proficiency level of the group; extra-classroom activities, such as: visits to museums, excursions, intercultural events, etc.); the role of the student in the group (a student who helps their colleagues a lot, who is a leader or one who does not display these qualities etc.).

4.2. Text and task metadata

Text and task metadata must also be well documented.

4.2.1. Task-related metadata

These are:

- date;
- institution,
- how many hours of class time have passed before the task had been set;
- how it had been elicited – a spontaneous production or one that had been prepared;
- whether the task had been inspired as the result of seeing a picture/video (yes/no answer); where images/photographs were used, these items will also be preserved in association with other relevant informative metadata;
- the use of sources of reference (dictionaries, textbooks, internet sources, etc., all kinds of sources);
- the type of task (an exercise that has been completed in class, something set as homework in the course of the working week, work to be done during the holidays, a monthly assignment, work done in an exam, etc.); the requirements laid down for completion of the task will also be recorded as informative metadata;
- time limitation / time bound – whether the student was given a deadline to hand in the exercise (timed or untimed; homework, fixed, free; defined subject / free choice);
- whether the student was required to produce something of a particular length (requirement as to length/no requirement);

- if a particular length was required, how many words (or replies, in the case of dialogues) were requested.

4.2.2. Language-related metadata

They are:

- general proficiency level of the group when the task was assigned (A1, A2, B1, B2, C1, C2);
- an oral or a written task;
- if oral, whether it is a monologue or a dialogue;
- if written, whether it was initially handwritten and subsequently word-processed or written directly using a computer; whether automatic spelling or automatic translation tools were used;
- if written, whether it is a dialogue or not;
- whether it is a general text or a text for a specific purpose;
- the style: narration, description, argumentation;
- the genre: message/e-mail/letter (formal/informal), fiction, diary, essay, academic text, etc.;
- topics: leisure, transportation, famous people, daily routines, etc.

5. Ethical issues

There are a range of serious ethical issues that arise from the particularities of our core activity, namely, the design and collection of a learner corpus, its storage and the data processing and post-processing. Such issues demand careful consideration (see also the proceedings of ETHI-CA2, 2016). It is, of course, well understood that both learners and researchers must comply with national laws regarding the processing of personal information and with the EU General Data Protection Regulation (EU-GDPR, 2016/679). The Research Ethics Committee[5] of the University of Bucharest is assisting us in drawing up the legal forms to be signed by the learners and the team members.

The learners involved will be invited to sign a form giving informed consent, whereby they give their agreement to written and oral samples of their productions being used by the LECOR project. Learners that participate in the project will agree that they will have no pecuniary claims or claims of any kind (arising, for example, from their rights as author of the samples created by them) as a result of the archiving and use of their work undertaken during the courses on the Preparatory Year and sampled for the LECOR corpus. The informed consent also sets out: a) the procedures adopted for ensuring data protection/confidentiality/privacy; b) information as to the voluntary nature of the decision to take part in the project; c) the right to withdraw at any time from it without consequences; d) potential risks; e) the expected duration of the subject's participation; f) the circumstances in which a third party can process a subject's personal data etc. (see also the European Commission's guidelines for Informed Consent on the CORDIS portal – Community Research and Development Information Service).

[5] https://cometc.unibuc.ro/

For their part, members of the project will sign a declaration, in which they guarantee not to disseminate information or sensitive details that might allow learners taking part to be individually identified. They will also guarantee to respect all aspects of the protection of natural persons in relation to the processing of personal data.

The particular activities to be undertaken by the project and which give rise to a variety of ethical problems are the following:

● The collection of written samples (by hand and subsequently word-processed or samples written directly using a computer). These samples are to be taken from the work of foreign students who are pursuing courses as part of the Preparatory Year at University of Bucharest. The work sampled will, in fact, be the product of such learners' regular assignments and activities, which form a normal part of their learning process (*e.g.* exercises undertaken in class or in exams, homework, etc.).

● The completion, with the participation of the individual learner, of a form which details the relevant data needed later on in order to compile the desired corpus and to allow research (*e.g.* native language, age, gender, proficiency level, etc.).

● The storage of such data on a server dedicated to this project. The server will be located in the University of Bucharest building and maintained by a university employee. The data stored will be the samples produced by the students (the initial written form, transcriptions of oral samples as well as audio recordings). It will also include the relevant accompanying information necessary in order to study the process of acquisition of Romanian as a foreign language (*i.e.*, the variables or metadata).

● The processing of texts and of the collected information about the learners and their productions (their anonymization and annotation, etc.).

● Making the corpus available to any interested parties, in open-access format, for the purposes of research (in order to allow searches).

● Providing the participants detailed information about the project and the manner in which their personal data will be protected; requesting and obtaining unconditional, informed consents from them in order to develop the activities implied by the objectives of the project. Such informed consent will result in a document being signed by each of the participants.

It is possible that, with reference to the context of the project, some of the learners taking part will be vulnerable individuals (*e.g.* children or people who are unwell, etc.). It is possible that such people may not understand completely what is being explained to them when their informed consent is sought. They may not comprehend what is being asked of them (see i, ii below). Additionally, they may become anxious if they are asked to sign a document (see iii below). Project members can anticipate and reduce these problems with vulnerable learners as follows:

(i) Where a student has insufficient understanding of Romanian, project members can explain in either English or French. If necessary, the explanations will also be translated into the learner's native language. The document giving consent will be forwarded to all participants in English as well as Romanian. Participants will be given sufficient time to study the document asking for informed consent, and to think carefully as to whether or

not to sign it. Participants will be told that they can, at any time, withdraw any consent they may have given and that this will not result in negative consequences for them.

(ii) Where a participant may not have the capacity to easily understand what is being said about the project and their part in it, project members will use clear concrete examples to explain at a level that will be understood by the student. They will use such examples to illustrate how personal data will be anonymized. They will explain that learner dictionaries and textbooks exist for the English language and that these have been created using a language corpus of samples of non-native students' work. Similar information needs to be collected about Romanian as a foreign language in order to improve the current textbooks and other learning materials. For example, exercises based on the errors made by non-native learners can be included in a textbook for future learners of Romanian as a second language or in a learner dictionary for Romanian.

(iii) If a given student is anxious when project members (their teachers, in fact) ask them to sign a document giving informed consent, project members will reinforce the explanations already given as to the implications of giving such a consent and the fact that they can withdraw the consent at any time. It will be pointed out that the benefit of their giving consent will be the improvement of the teaching and learning of Romanian as a foreign language.

As for the pseudonymisation of the data, all information about the author of written or oral samples (name, surname, telephone number, other forms of identification numbers, etc.) will be anonymized, so that these personal data cannot be attributed to the particular author. To this end, each learner and each text will be given a code. With the agreement of participants, information as to their age will also be recorded, as age is an important variable in the acquisition of a (foreign) language.

In anticipation of possible problems relating to gender discrimination, on forms containing sociolinguistic data (which will become metadata) rubrics as to gender/sex will contain a third option ('other – please fill in, if you want') and a fourth one such as 'I prefer not to say', in addition to the options masculine and feminine.

6. The importance of the LECOR project

The LECOR corpus will have many possible end-uses in language teaching and in natural language processing (for typical applications of this type of corpora, see McEnery *et al.*, 2006; O'Keeffe *et al.*, 2007; Römer, 2008; Granger, 2008; Díaz-Negrillo and Thompson, 2013; Granger *et al.*, 2015). LECOR's scalability will permit the subsequent development of an increasingly large annotated corpus focused on Romanian. This fact is very important, because the larger a digitalised corpus, the more reliable it is for a wider range of applications.

Acknowledgements

This work was supported by a grant from the Romanian Ministry of Education and Research, CNCS - UEFISCDI, project number PN-III-P1-1-1.1-TE-2019-1066, within PNCDI III.

References

Arieșan, A. and Vasiu, L.-I. (2016). On the necessity of a learner corpus – Romanian as a Foreign Language (RFL). In Boldea, I. (ed.). *The Proceedings of the International Conference Globalization, Intercultural Dialogue and National Identity*, 3. Tîrgu-Mureş, Arhipelag XXI Press, 955-966.

Bocoș, C. (2017). The complexity and accuracy of noun phrases with modifiers in written productions of learners of Romanian as a foreign language. *Studia Universitatis Babes-Bolyai. Philologia*, 62, 2, 79-92.

Brook-Hart, G. (2009). *Learning from Common Mistakes*. Cambridge, Cambridge University Press.

Callies, M. (2015). Learner corpus methodology. In Granger *et al.* (eds) 2015, 35-55.

Chaudron, C. (2003). Data collection in SLA research. In Doughty, C. J., Long, M. H. (eds). *The Handbook of Second Language Acquisition.* Malden, MA, Blackwell, 762-828.

Constantinescu, M.-V, Stoica, G. (2020). *Româna ca limbă străină. Corpus*. București, Editura Universității din București.

Díaz-Negrillo, A., Thompson, P. (2013). Learner corpora. Looking towards the future. In Díaz-Negrillo, A., Ballier, N., Thompson, P. (eds). *Automatic Treatment and Analysis of Learner Corpus Data*. Amsterdam/Philadelphia, John Benjamins Publishing Company, 9-30.

Dörnyei, Z. (2005). *The Psychology of the Language Learner. Individual Differences in Second Language Acquisition.* Mahwah, NJ, Lawrence Erlbaum.

ETHI-CA2 2016 = Proceedings of ETHics. In *Corpus Collection, Annotation & Application*, LREC, Portorož, Slovenia, 2016.

Gafu, C., Badea, M., Iridon, C. (2012). Errors in the acquisition of Romanian as second language. A case study. *Procedia – Social and Behavioral Sciences*. 69, 1626-1634.

Geană, I. (2018). Limba română ca limbă străină. In Sala, M., Saramandu, N. (eds), *Lingvistica românească*. București, Editura Academiei Române, 687-699.

Gilquin, G. (2015). From design to collection of learner corpora. In Granger *et al.* (eds) 2015, 9-34.

Granger, S. (2002). 'A Bird's–eye view of learner corpus research. In Granger, S., Hung, J., Petch-Tyson, S. (eds). *Computer Learner Corpora, Second Language Acquisition and Foreign Language Teaching*. Amsterdam/Philadelphia, John Benjamins Publishing Company, 3-33.

Granger, S. (2008). Learner corpora in foreign language education. In Van Deusen-Scholl, N., Hornberger, N. (eds), *Encyclopedia of Language and Education*, vol. 4, *Second and Foreign Language Education*. Springer Netherlands, 337-351.

Granger, S., Gilquin, G. and Meunier, F. (eds) (2015). The Cambridge Handbook of Learner Corpus Research. Cambridge, Cambridge University Press.

Granger, S., Gilquin, G. and Meunier, F. (2015). Introduction: learner corpus research – past, present and future. In Granger *et al.* (eds) 2015, 1-6.

Granger, S. and Paquot, M. (2017). Towards standardization of metadata for L2 corpora. *CLARIN workshop on Interoperability of Second Language Resources*

and Tools (Göthenburg, Sweden, 06-08/12/2017): http://hdl.handle.net/2078.1/198216.

Ion, R. (2018). TEPROLIN: An Extensible, Online Text Preprocessing Platform for Romanian. In *Proceedings of the International Conference on Linguistic Resources and Tools for Processing Romanian Language (ConsILR 2018)*, November 22-23, 2018, Iași, România, 69-76.

Lüdeling, A. and Hirschmann, H. (2015). Error annotation systems. In Granger *et. al.* 2015 (eds), 135-158.

Maiden, M. (2016). Romanian. In Ledgeway, A., Maiden, M. (eds), *The Oxford Guide to the Romance Languages*. Oxford, Oxford University Press, 91-125.

McEnery, T., Xiao, R., Tono, Y. (2006). *Corpus-based Language Studies: An Advanced Resource Book*. London/New York, Routledge, 97-103.

Mîrzea Vasile, C. (2016). I più recenti manuali di rumeno come lingua straniera. *Romània orientale*, XXIX, 287-310.

Mîrzea Vasile, C. (2017). Corpusurile de limba română și importanța lor în realizarea de materiale didactice pentru limba română ca limbă străină. *Romanian Studies Today*, 1, 74-95.

O'Keeffe, A., McCarthy, M., Carter, R. (2007). *From Corpus to Classroom: Language use and Language Teaching*. Cambridge, Cambridge University Press.

Pravec, N. A. (2002). Survey of learner corpora. *ICAME Journal*, 26, 81-114.

Rastelli, S. (2009). Learner corpora without error tagging. *Linguistik Online*, 38, 2, 57-66.

Römer, U. (2008). Corpora and language teaching. In Lüdeling, A., Merja, K. (eds). *Corpus Linguistics. An International Handbook*, vol. 1, Berlin, Mouton de Gruyter, 112-130.

Rundell, M. (2007). *MacMillan English dictionary advanced learner*. MacMillan.

Schmitt, D. and Schmitt, N. (2011). *Focus on Vocabulary 2: Mastering the Academic Word List, White Plains*. NY: Pearson Education.

Tono, Y. (2016). What is missing in learner corpus design?. In Alonso-Ramos, M. (ed.), *Spanish Learner Corpus Research. Current Trends and Future Perspectives.* Amsterdam/Philadelphia, John Benjamins Publishing Company, 33-52.

Van Rooy, B. (2015). Annotating learner corpora. In Granger et. al. 2015 (eds), 79-106.

Vasiu, L.-I. (2020). *Achiziția limbii române ca L2. Interlimba la nivelul A1*. Cluj-Napoca, Presa Universitară Clujeană.

BEGINNING AND END OF SENTENCE WORD DIGRAMS FOR PRINTED ROMANIAN LANGUAGE

ALEXANDRU DINU[1], ADRIANA VLAD[1,2], ADRIAN MITREA[1]
AND BOGDAN HANU[1]

[1]*Faculty of Electronics, Telecommunications and Information Technology, POLITEHNICA University of Bucharest, alexandrudinu89@yahoo.com*
[2]*The Research Institute for Artificial Intelligence, Romanian Academy avlad@racai.ro, adriana_vlad@yahoo.com*

Abstract

A detailed experimental study was conducted concerning digrams of words beginning and ending sentences in the Romanian language. The investigation was done on a literary corpus of about 6.3 million words, in printed form with orthographical characters and punctuation marks (alphabet of 47 symbols). The sentences are separated by a word which contains at least one of the 4 separator symbols: full stop, question mark, exclamation mark or ellipsis.

One first perspective of the analysis led to an overall evaluation of the structure of the two successive words (word digrams) beginning and ending a sentence or complex sentence on the entire corpus (49 books) and on some subcollection of works. Aspects related to the impact of orthography and punctuation in natural language were taken into consideration.

The analysis continued by searching along the Zipf's rank-frequency law the individual words which form the most frequent digrams beginning and ending sentences. The results showed a very good coverage of the beginning of sentence digrams with words from Zipf first area of ranks of priority interest for the natural language user.

It was also identified the intersection between the sets of the distinct digrams beginning and ending sentences with the set of common word digrams from all the books in the corpus. More than 90% of the common digrams can appear in the beginning of sentence while only 1% can build the end of sentence.

Keywords —beginning and end of sentence, word digrams, Zipf's Law

1. Introduction

This paper continues the authors' analysis regarding the sentence structure in the Romanian language. As part of the previous investigations, results regarding words in general and beginning and end of sentence words' structure were presented in (Hanu *et al.*, 2018), (Hanu *et al.*, 2019). As a novel element, this paper presents results regarding the beginning and end of sentence word digrams (groups of two consecutive words).

The experiments are made on a literary corpus (novels and short stories) of over 6.3 million words in length and made up of 49 books from 9 writers: Isaac Asimov – 9 books, Constantin Chiriţă – 5 books, Alexandre Dumas – 12 books, Colin Falconer – 1

book, Frank Herbert – 8 books, Niven Larry – 1 book, Orson Scott Card – 3 books, Michel Zevaco – 7 books, J. R. Tolkien – 3 books. The corpus was used in previous analyses of the authors and is presented in great detail in (Vlad *et al*., 2011), (Vlad *et al*., 2013), (Hanu *et al*., 2016) and (Hanu *et al*., 2018).

The extended alphabet consists of 47 symbols: the 31 corresponding to the basic set of letters: A Ă Â B C D E F G H I Î J K L M N O P Q R S Ş T Ţ U V W X Y Z, the blank and 15 orthography and punctuation marks: hyphen, full stop, comma, colon, semicolon, question mark, exclamation mark, quotation dash, em dash, abbreviation point, ellipsis, quotation marks, parentheses and apostrophe.

A word is defined as the sequence of characters between two spaces/blanks, so the punctuation marks are attached to the word. The dialogue appears as a distinct word in the corpus (denoted by *{*). When using the 47-symbol extended alphabet (orthography and punctuation are integral part of the word) all below constructions are different words:

SPUS SPUS, SPUS. SPUS?

SPUS: SPUSE SPUSE: SPUSESE

In the authors' previous papers (Hanu *et al.*, 2018; Hanu *et al.*, 2019), what was considered as the end of sentence word was the word that contains on its last position before blank one of the 4 symbols from the extended alphabet: Full Stop, Question Mark, Exclamation Mark, Ellipsis (which means that only the situations when the 4 elements were followed by a blank were considered in the analysis). The word that followed one of the 4 situations mentioned above was considered as beginning of sentence word. The total number of sentences calculated with this rule was 528,976.

Nevertheless, there are scenarios when the sentences end with one of the 4 punctuation symbols mentioned above and, before the blank, there is a different character, like parentheses or quotation marks. The quotation marks and parentheses contribute with 3671 and 1055 new sentences, respectively. This type of situations has been considered in the present paper, leading to an improved accuracy of the results. Consequently, the total number of sentences based on the new rule (in determining the beginning and end of sentence) has become 533,988. The word digrams' analysis took into account only the sentences of length at least two (otherwise a word digram cannot be formed). The number of sentences of length 1 is 11,413, so we have 533,988 – 11,413 = 522,575 sentences of length greater or equal than 2, thus 522,575 beginning and end of sentence digrams.

The purpose of the first subanalysis which follows is to better understand and count the end of sentence words which include more than one end of sentence symbol. Most of the sentences from the corpus end by a word which contains only one of the four separators (99% – 528,374 sentences), approximately 1% of the sentences end by two of the four separators (5,552 sentences, more precisely) and a minority of sentences end by three of the four separators (0.01% – 62 sentences).

Table 1 shows the end of sentence words for the sentences ended by three out of the four separators and a specific sentence example is also presented next. An example of

sentences in the corpus is the following: “CINE SĂ AIBĂ BARCĂ ÎN ORAȘ? DE CE SĂ AIBĂ?”.

Table 1: End of sentence words containing 3 end of sentence separators

CEASUL?!}	OFER!}.	CAVALERE!}.
SERGIU?!}	VINĂ?!}	LOR!}.
DAAAAA!?}	CRÂȘMĂ!?}	MINE!}.
NEBUN?!}	MAMA!}.	MILĂ!}.
FRICĂ?!!}	CRĂCIUN!}.	NOROI}?!
ȘAH!?}	ÎNȘELAT?}.	LABA?!}
GRANIT?!}	MINE!}.	UNA!}.
ZĂU!?}	MARILLAC!}.	URSULE?}.
CĂLĂTORIE?!}	COMANDE!}.	RĂZBOI?!}
CURSĂ?!}	LITURGHIA!}".	SPAȚIALI?!.
CE?}OH!	BUCUROS?}.	MORT!}.
ALLAN?!}	RĂU!}.	RIIILFY!}.
VRUT!"}?	CĂLUGĂR?}".	MOARTE!}.
CE.?}	TEMPLUL!}.	ACOLO?!}
SAU.?}	CALE!}.	EI?!}
ROSPIGNAC!}.	SÂNGE!}.	SULINA}!?
NAIBA?!}"	EU!}.	T.?!
REGELE!}.	VIU!}.	PUTERI!?}
MOARTEEEEE!}.	GĂSESC!}.	PUTINȚĂ?!}
MOARTE!}.	MARILLAC!}.	IDEE!}.
FACE!}.	OFER!}.	EȘTI!}.

It is important to notice that the consideration of writing with orthographical characters and punctuation marks made possible the analysis of words and group of words beginning and ending sentences/complex sentences in the natural language. To our knowledge, the orthography and punctuation elements have not been enough investigated neither for Romanian, nor for other natural languages, at least not in the sense of being included in the mathematical language description (Say and Akman, 1996; Rodriguez-Castro, 2011; Vlad *et al*., 2013; Hanu *et al*., 2018).

2. Beginning of sentence word digrams

The first part of the analysis takes into account the overall corpus of 49 books and will focus on the beginning of sentence word digrams. Table 2 presents the most frequent 10 beginning of sentence word digrams depending on the end of sentence separator/symbol of the previous sentence. We considered here only the sentences ended by one single separator which could be followed either by space, parentheses + space or quotation marks + space. The number of distinct beginning of sentence word digrams is 184 337.

Table 2: Beginning of sentence word digrams

	Any separator			Full Stop			Question mark		
0	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
1	{	NU	1.09	{	ȘI	0.75	{	DA,	0.26
2	{	ȘI	0.90	{	NU	0.73	{	NU	0.24
3	{	DA,	0.73	{	DAR	0.47	{	NU,	0.16
4	{	DAR	0.57	{	DA,	0.42	{	DA.	0.11
5	{	CE	0.49	{	CE	0.39	{	DE	0.10
6	{	DE	0.49	{	DE	0.34	{	PENTRU	0.09
7	{	NU,	0.41	{	AM	0.25	ÎL	ÎNTREBĂ	0.09
8	{	AM	0.37	{	E	0.23	{	AM	0.09
9	{	ÎN	0.34	{	ÎN	0.23	{	ÎN	0.08
10	{	E	0.32	{	EI	0.22	ÎNTREBĂ	EL.	0.07

Table 2: Beginning of sentence word digrams (continued)

	Exclamation mark			Ellipsis		
0	**1**	**2**	**3**	**1**	**2**	**3**
1	{	NU	0.05	{	NU	0.06
2	{	ȘI	0.05	{	ȘI	0.04
3	{	DAR	0.03	{	DA,	0.03
4	{	DA,	0.03	{	DAR	0.02
5	{	CE	0.03	{	CE	0.02
6	STRIGĂ	EL.	0.02	{	DE	0.02
7	**{**	**DE**	**0.02**	DAR	NU	0.02
8	ÎȘI	SPUSE	0.02	{	SĂ	0.02
9	DE	CE	0.02	{	EI	0.02
10	SPUSE	EL.	0.02	PENTRU	CĂ	0.02

The distinct word digrams occurring at the beginning of a sentence can also appear in the middle of a sentence and, by considering all their occurrences in the corpus, one can calculate that they cover the corpus in a proportion of 38.3%; strictly considering them as used at the sentence beginning, they cover the corpus only in proportion of

approximately 8% (ratio of the number of sentences of length larger than 2 to the corpus length in digrams).

How to read Table 2: Column 0 represents the rank of the beginning of sentence word digram, columns 1 and 2 contain the words which compose the word digram and column 3 represents the relative frequency (in %) of the specific word digram with respect to the total number of sentences with length larger than 2 words. Columns 1, 2 and 3 are replicated for different end of sentence separators. Row 7 from Table 2 shows the results for the 7th most frequent word digram depending on the end of sentence symbol for the previous sentence. For example, the 7th most frequent beginning of sentence word digram in the corpus following after a sentence ended by exclamation mark is ***{ DE***, which appears in the beginning of **0.02%** of the sentences of the corpus.

Based on the experimental data presented in Table 2, one can easily notice that the first words of the beginning of sentence word digrams are a lot of words which themselves are very frequent in the corpus. For example, the dialogue mark (represented by the symbol ***{)*** and the word ***DE*** are in top 3 of the most frequent words in the corpus.

If we consider only the sentences ended by a single end of sentence separator (99% of the sentences), we obtain the following inventory:

1. 389,284 sentences end by Full Stop (74% from the total number of sentences of length larger than two)
2. 60,390 sentences end by Question Mark (11.5% of the sentences)
3. 39,995 sentences end by Exclamation Mark (7.7% of the sentences)
4. 27,767 sentences end by Ellipsis (5.3% of the sentences)

As one can easily notice, there are approximately 1% of the sentences unaccounted for and those are the sentences ended by two or three end of sentence symbols.

2.1. Word digrams beginning sentences and Zipf's law

Previous papers of the authors delivered multiple results regarding the separation of words into three areas of priority interest for natural language researchers, depending on the number of occurrences of the distinct words in the corpus. For the overall corpus (49 books), Area 1 (established along the rank axis of the Zipf's law) corresponds to words with more than 200 occurrences. These are the first 2762 most frequent distinct words and cover approximately 72% of the corpus. We were interested in how many beginning of sentence word digrams contain words from Area 1. The results obtained are the following: over 67% of the beginning of sentence word digrams contain both words from Area 1, approximately 28.7% contain one word from Area 1 and 4% of the digrams appearing at the beginning of sentences do not contain any word from Area 1.

2.2. Word digrams beginning sentences and common word digrams in the corpus

Another perspective was focusing on the relation between the beginning of sentence word digrams and the common ones in the corpus, *i.e.*, the word digrams which appear identically written in all 49 books. Previous research of the authors led to the creation of

the list of the 578 word digrams common to the entire corpus. Approximately 90% of the them were identified as beginning of sentence word digrams.

2.3. Word digrams beginning sentences – comparative view between the overall corpus and a subcorpus

Another view on the beginning and end of sentence word digrams was to analyse a subcorpus with books from 3 different authors (Chirita, Herbert and Dumas) and to compare the results with the data obtained from the overall corpus analysis. The total number of sentences of the subcorpus (20 books) is 319,098 (approximately 60% from the total number of sentences of the overall corpus). 7,612 of these sentences are not considered in the subsequent analysis because they have a length of one word. Table 3 shows the most frequent 15 beginning of sentence word digrams, independent on the end of sentence symbol of the previous sentence. This analysis is performed for the two scenarios: the overall corpus and the subcorpus. One can notice a very good stability of the results between the overall corpus and the subcorpus (from a rank and relative frequency perspective; the relative frequencies are defined with respect to the total number of sentences from each analysed case/corpus). The lines which are highlighted are the ones where small differencies are present.

How to read Table 3: Row 15 contains the beginning of sentence word digram DE CE (columns 0 and 1), which has rank 15 (column 2) in the hierarchy of beginning of sentence word digrams from the overall corpus from a decreasing probability point of view and rank 20 based on the same principles in the subcorpus (column 3). The relative frequency of the digram in the overall corpus is 0.23% (column 4) and is obtained as the ratio between the number of occurrences and the total number of sentences in the corpus (1207 / 522,575 = 0.23%). Column 5 contains the relative frequency of the same word digram in the subcorpus (0.22% = 696/311,486).

Table 3: The first 15 most frequent beginning of sentence word digrams independent on the end of sentence separator symbol. 0. First word; 1. Second word; 2. Rank in the beginning of sentence word digrams' hierarchy – overall corpus; 3. Rank in the beginning of sentence word digrams' hierarchy – subcorpus; 4. Relative frequency – overall corpus (%); 5. Relative frequency – subcorpus (%).

0	1	2	3	4	5
{	NU	1	2	1.09	0.97
{	ȘI	2	1	0.90	1.05
{	DA,	3	3	0.73	0.87
{	DAR	4	4	0.57	0.60
{	CE	5	5	0.49	0.55
{	DE	6	6	0.49	0.49
{	NU,	7	7	0.41	0.41
{	AM	8	8	0.37	0.39
{	ÎN	9	10	0.34	0.36
{	E	10	9	0.32	0.37
{	EI	11	11	0.29	0.35
{	SĂ	12	12	0.27	0.31
{	AI	13	13	0.26	0.29
{	O	14	17	0.24	0.25
DE	**CE**	**15**	**20**	**0.23**	**0.22**

3. End of sentence word digrams

Table 4 shows the most frequent end of sentence digrams depending on the end of sentence separator. Column 0 represents the rank of the end of sentence word digram, columns 1 and 2 contain the words which compose the word digram and column 3 represents the relative frequency (in %) of the specific word digram with respect to the total number of sentences with length larger than 2 words. Columns 1, 2 and 3 are replicated for different end of sentence separators. Row 7 from Table 4 shows the results for the 7th most frequent end of sentence word digram depending on the end of sentence symbol. For example, the 7th most frequent end of sentence word digram in the corpus independent on the end of sentence symbol is ***ÎNTREBĂ EL.***, which appears at the end of **0.09%** of the sentences of the corpus.

Table 4: End of sentence word digrams

	Any separator			**Full Stop**			**Question mark**		
0	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
1	NU-I	AȘA?	0.20	{	DA.	0.16	NU-I	AȘA?	0.20
2	{	DA.	0.16	SPUSE	EL.	0.16	DE	CE?	0.13
3	SPUSE	EL.	0.16	DIN	CAP.	0.14	CE	NU?	0.04
4	DIN	CAP.	0.14	ZISE	EL.	0.10	S-A	ÎNTÂMPLAT?	0.03
5	DE	CE?	0.13	ÎNTREBĂ	EL.	0.09	{	EU?	0.03
6	ZISE	EL.	0.10	DE	EL.	0.09	{	CUM?	0.03
7	**ÎNTREBĂ**	**EL.**	**0.09**	{	NU.	0.09	PENTRU	CE?	0.03
8	DE	EL.	0.09	AȘA	CEVA.	0.07	SĂ	SPUI?	0.03
9	{	NU.	0.09	MAI	MULT.	0.07	CU	MINE?	0.02
10	AȘA	CEVA.	0.07	SPUSE	EA.	0.07	AȘA	CEVA?	0.02

Table 4: End of sentence word digrams (continued)

	Exclamation mark			**Ellipsis**		
0	**1**	**2**	**3**	**1**	**2**	**3**
1	{	AH!	0.06	{	DA}	0.02
2	{	NU!	0.05	{	DAR}	0.02
3	{	OH!	0.04	{	NU}	0.01
4	LA	NAIBA!	0.03	{	}	0.01
5	{	AHA!	0.03	NU	ȘTIU}	0.01
6	{	DA!	0.03	ȘI	TOTUȘI}	0.01
7	LA	DRACU!	0.03	CINSTEA	MEA}	0.01
8	TOȚI	DRACII!	0.02	{	PĂI}	0.01
9	FOARTE	BINE!	0.02	PENTRU	CĂ}	0.01
10	{	DRACE!	0.02	AȘA	CĂ}	0.01

It is important to mention that there are 289,894 distinct end of sentence word digrams, a much larger number compared to the beginning of sentence word digrams (1.5 times more than the 184,337 distinct beginning of sentence digrams). We could say that the beginning of sentence follows a pattern much more than the end of sentence does (a lot of frequent words are part of the beginning of sentence, the sentences start very often by dialogue, etc.), while the end of sentence is much more diverse.

3.1. End of sentence word digrams and Zipf's law

Next, we will follow a similar approach as in Subsection 2.2. This time we are interested in the end of sentence digrams and how many of them contain words from Area 1 established along the rank axis of the Zipf's law. The results are the following: more than 19% of the end of sentence digrams contain both words from Area 1, approximately 64.4% have only one word from Area 1 and 16.5% from the end of sentence word digrams do not contain any word from Area 1. One can notice that the results are different compared to the beginning of sentence analysis: the digrams from the beginning of a sentence contain both words from Area 1 in a higher percentage compared to the end of sentence digrams (67% vs 19%).

3.2. End of sentence word digrams and common digrams from the corpus

If we switch again the perspective and are interested in how many end of sentence word digrams are also common digrams for all the books in the corpus, the percentage is only 1%. The result is much smaller compared to the beginning of sentence digrams (90%), another proof on the higher variability of end compared to beginning of the sentence.

3.3. End sentence word digrams – comparative view between the overall corpus and a subcorpus

Table 5 shows the most frequent 15 end of sentence word digrams independent on the end of sentence symbol. This analysis is performed for two scenarios: the overall corpus and the subcorpus introduced in Subsection 2.4.

Table 5: The first 15 most frequent end of sentence word digrams independent on the end of sentence symbol 0. First word; 1. Second word; 2. Rank in the end of sentence word digrams' hierarchy – overall corpus; 3. Rank in the end of sentence word digrams' hierarchy – subcorpus; 4. Relative frequency – overall corpus (%); 5. Relative frequency – subcorpus (%).

0	1	2	3	4	5
NU-I	AȘA?	1	1	0.20	0.19
{	DA.	2	2	0.16	0.18
SPUSE	EL.	3	3	0.16	0.15
DIN	CAP.	4	4	0.14	0.12
DE	CE?	5	6	0.13	0.10
ZISE	EL.	6	7	0.10	0.09
ÎNTREBĂ	EL.	7	5	0.09	0.10
DE	EL.	8	8	0.09	0.08
{	NU.	9	9	0.09	0.07
AȘA	CEVA.	10	46	0.07	0.04
MAI	MULT.	11	13	0.07	0.06
SPUSE	EA.	12	11	0.07	0.07
CU	EL.	13	16	0.07	0.06
DIN	UMERI.	14	15	0.07	0.06
DE	**EA.**	**15**	**21**	**0.06**	**0.05**

One can notice a good stability of the results between the overall corpus and the subcorpus (from a rank and relative frequency perspective; the relative frequencies are defined with respect to the total number of sentences from each scenario analysed). The lines which are highlighted are the ones where differencies are present. The end of sentence digrams seem more sensitive from a rank/relative frequency perspective compared to the beginning of sentence digrams (more rank differencies in Table 5 compared to Table 3). The punctuation marks, which contributes to the originality of an author, could explain these results.

How to read Table 5: Row 15 contains the end of sentence word digram DE EA. (columns 0 and 1), which has rank 15 (column 2) in the decreasing hierarchy of end of sentence word digrams from the overall corpus and rank 21 based on the same principles in the subcorpus (column 3). The relative frequency of the digram in the overall corpus is 0.06% (column 4) and is obtained as the ratio between the number of occurrences and the total number of sentences in the corpus (331/522,575 = 0.06%). Column 5 contains the relative frequency of the same word digram in the subcorpus (0.05% = 152/311,486).

4. Conclusions

Based on the results presented so far, we can draw some general conclusions about the beginning and end of the sentence word digrams. The number of distinct beginning of sentence digrams is lower than the number of distinct end of sentence digrams (184,337 versus 289,894). We could say that the beginning of the sentence is somehow more predictable (expected) compared to the end of sentence. Frequent words in the corpus appear with predilection in the structure of the beginning of sentence word digrams (see the Zipf's law discussion). The common word digrams in the corpus are found to a greater extent (90%) as digrams at the beginning of the sentence than (1%) as digrams at the end of the sentence.

About 99% of the sentences end by only one end of sentence separator. Among these, approximately 75% end with a full stop, the rest of the sentences ending by question or exclamation mark or ellipsis. There are also sentences ending in two or three end of sentence symbols but these sentences form a minority. The dialogue is predominant in the composition of the beginning of the sentence digrams and also appears in the end of the sentence digrams (for sentences of length 2). Note that the dialogue sign is a standalone word in the corpus.

The second word in the end-of-sentence diagrams contains at least one punctuation element, marking the separation into sentences.The end-of-sentence digrams cover the corpus in a proportion of about 8%. The distinct word digrams which are found at the beginning of a sentence can also appear in the middle of a sentence and, by considering all their occurrences in the corpus, they cover the corpus in a proportion of 38.3%; strictly considering them as used at the beginning of sentence, they cover the corpus only in a proportion of about 8% (same as for the end of sentence word digrams).

Looking at the overall results, the paper emphasizes once more the impact/role of orthography and punctuation in the language model and the importance of beginning and end of sentence word diagrams in the statistical description of the language.

Acknowledgment

The authors acknowledge the continuous scientific support from Prof. Dan Tufiș, member of the Romanian Academy. The work is part of an ongoing research theme at the Research Institute for Artificial Intelligence, Romanian Academy.

References

Hanu, B., Vlad, A., Mitrea, A. and Dragomir, R. (2016). An analysis of common word digrams in different literary Romanian corpora. In *2016 International Conference on Communications (COMM)*, Bucharest, 43-46.

Hanu, B., Vlad, A. and Mitrea, A. (2018). Aspects Revealing the Orthography and Punctuation Impact in Printed Romanian: A Literary Corpus Based Study. In *2018 International Conference on Communications (COMM)*, Bucharest, 2018, 95-100.

Hanu, B., Vlad, A., Dinu, A. and Mitrea, A. (2019). Looking Along Zipf's Law for the Distribution of Words Beginning and Ending Sentences in Literary Printed Romanian Corpora. In *Proceedings of the 14th International Conference "Linguistic Resources and Tools for Natural Language Processing"*, Cluj-Napoca, 18-20 November 2019, 51-62.

Rodríguez-Castro, M. (2011). Translationese and punctuation: An empirical study of translated and non-translated international newspaper articles (English and Spanish). *Translation and Interpreting Studies*, 6, 1, 40-61.

Say, B. and Akman, V. (1996). Current approaches to punctuation în computational linguistics. *Computer and the Humanities*, 30, 457-469.

Vlad, A., Mitrea, A., Ciucă, Șt. and Luca, A. (2011). A study on the statistical structure of words and of word digrams in a literary Romanian corpus. In *6th Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, Brasov, 1-8.

Vlad, A., Mitrea, A., Luca, A. and Hodea, O. (2013). Considerations Regarding the Statistical Compatibility of Two Romanian Literary Corpora with Orthography and Punctuations Marks Included. In D. Tufis, V. Rus, C. Forascu (eds), *Towards Multilingual Europe 2020: A Romanian Perspective*, The Publishing House of the Romanian Academy, 99-122.

CHAPTER 2.
TOOLS FOR
NATURAL LANGUAGE PROCESSING

MULTIPLE ANNOTATION PIPELINES INSIDE THE RELATE PLATFORM

VASILE PĂIȘ

Research Institute for Artificial Intelligence, Romanian Academy

vasile@racai.ro

Abstract

RELATE is a modular high-performance platform integrating different state of the art algorithms for natural language processing tasks. Initially developed within the ReTeRom project by integrating the TEPROLIN web service, as well as other technologies, it was expanded in the MARCELL project by integrating IATE and EuroVoc annotators into a single complex pipeline. The present work describes an extension of the platform to allow multiple complete annotation pipelines to be executed and combined as needed with additional modules. This is exemplified by presenting the integration of UDPipe inside the RELATE platform. Furthermore, this opens up the possibility to annotate, inside the platform, multilingual corpora, with other languages apart from Romanian.

Key words — annotation pipelines, high-performance platform, natural language processing

1. Introduction

In the context of the ReTeRom[1] project, the RELATE platform (Păiș *et al*., 2019; Păiș *et al.*, 2020) was initially constructed as a way to interact with the TEPROLIN web service (Ion, 2018). This allowed for easy interaction with a complex pipeline for natural language processing of Romanian text. Furthermore, the high-performance design of the platform permits the distribution of annotation jobs to multiple servers, thus reducing the overall time required for annotating large corpora. It was further enhanced by integrating different query interfaces to the CoRoLa[2] corpus (Tufiș *et al*., 2019), including textual search using KorAP (Banski *et al*., 2012) and searching in the speech component (Boroș *et al*., 2018b). Even more, links to the Romanian WordNet (Tufiș and Barbu Mititelu, 2014) and identification of similar words based on word embeddings (Păiș and Tufiș, 2018) are integrated, in order to allow the user to explore these resources starting from annotated text. Another integration consists of an automatic speech recognition (ASR) system (Avram *et al.*, 2020) developed within the ROBIN[3] project, with better performance (Word Error Rate, Character Error Rate and mainly response time) than the previous one reported by Ion *et al.* (2020).

New annotation options were incorporated in the RELATE platform as part of the MARCELL[4] project, whose final goal is to deliver clean, validated and cross-lingual thematically clustered domain-specific language resources based on the national

[1] http://www.racai.ro/p/reterom/
[2] http://corola.racai.ro/
[3] http://aimas.cs.pub.ro/robin/en/
[4] https://marcell-project.eu/

legislation (laws, decrees, regulations) of the seven involved countries: Bulgaria, Croatia, Hungary, Poland, Romania, Slovakia and Slovenia. Thus, the project will aid in enhancing the automatic translation system eTranslation[5] of Connecting Europe Facility (CEF)[6] for the corresponding seven EU languages. For the purposes of this project, all the documents are annotated using EuroVoc[7] and IATE[8] (Interactive Terminology for Europe) descriptors. The annotation mechanism for the Romanian language[9] (Coman *et al.*, 2019) was integrated in the RELATE platform as an optional step in the processing pipeline. This allowed for the entire Romanian MARCELL legislative corpus (containing over 375 million tokens) (Varadi *et al.*, 2020; Tufiș *et al.*, 2020) to be fully annotated inside the platform. Furthermore, in order to comply with MARCELL's specific requirements, export filters were developed and integrated, allowing annotated text to be exported in both CoNLL-U Plus[10] and XML formats.

By annotating a large corpus, such as the MARCELL legislative corpus, it became apparent that the current pipeline is monolithic with some components waiting for others without making use of their provided output. For example, the EuroVoc annotation uses only word form and lemma, but was waiting for the entire TEPROLIN processing, which produces also part-of-speech, named entities and dependency parsing. Even more, this makes use internally of NLP-Cube (Boroș *et al*., 2018a), which has high annotation time (Alves *et al.*, 2020). Even though the overall annotation time was reduced by making use of multiple instances, spread across different servers, the annotation process took about one month for the entire MARCELL corpus. This seemed to justify looking for alternate pipelines that could provide sufficient data for later stages, but in a reduced amount of time, while still keeping the full TEPROLIN pipeline as an option when needing all the available annotations.

Furthermore, the more recent CURLICAT[11] project aims to compile curated datasets in seven languages of the consortium (Bulgarian, Croatian, Hungarian, Polish, Romanian, Slovak and Slovenian), in domains of relevance to European Digital Service Infrastructures (DSIs), with a view to enhancing the eTranslation automated translation system. The monolingual corpora, containing each at least 1 million sentences, will cover the following specific domains: finance, health, scientific research, cultural heritage, education, economics, politics. Collected data, having ensured IPR clearance and anonymization of the personal data in texts, will be extensively processed and aggregated with linguistic meta-information: PoS annotation and lemmatization; enrichment of monolingual data with IATE terms from the database of EU inter-institutional terminology; aggregation of monolingual data with domain-specific terms. From the perspective of annotating Romanian texts, this involves the integration of new modules in the RELATE platform, with different input requirements. Therefore, the ability to combine different pipelines and modules into a coherent annotation process

[5] https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eTranslation
[6] https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/CEF+Digital+Home
[7] https://eur-lex.europa.eu/browse/eurovoc.html
[8] https://iate.europa.eu/home
[9] https://github.com/racai-ai/IATE-EUROVOC-Annotator
[10] https://universaldependencies.org/ext-format.html
[11] https://curlicat-project.eu/

becomes extremely important in order to bridge the gap between the MARCELL project's work and the CURLICAT's envisaged actions.

The single processing pipeline approach for a given language was used in other platforms as well such as GATE (Cunningham, 2002). Even though the GATE architecture allows development of plugins for multiple pipeline components, the current implementation offers a single choice for any of those in a given language. Nevertheless, there are different modules available with similar functionality for different languages. This architecture was further developed in a more recent platform: TextFlows (Perovšek *et al.*, 2016). In this platform, multiple components with similar functionality are available, but the user has to manually integrate them into an annotation pipeline (known as a workflow in the platform). This adds complexity and requires the user to be familiarized with the platform itself and with the specific components (inputs from one component must be matched with outputs from other components). Furthermore, the user must provide training corpora for the majority of the components in order to enable annotation in languages other than English. Additionally, known language specific pre-trained annotators, such as TEPROLIN, UDPipe or TTL (Ion, 2007) for Romanian language, are not available in this platform, thus making it difficult to estimate the errors introduced by the modules used.

Ferrucci and Lally (2004) introduced the UIMA framework for unstructured information processing at IBM. It makes use of components called "Text Analysis Engines" which are similar to the processing resources used in GATE. One novelty is the introduction of the "Common Analysis Structure" (CAS) which represents a common format for storing data. However, this is mainly intended for developers, offering a number of interfaces in Java and C++ and is not directly accessible by the end-user via a web interface.

The ALPE platform (Cristea and Pistol, 2008; Pistol, 2011) makes use of a graph-like structure, also known as a hierarchy of annotation schemas (Cristea and Butnariu, 2004), for integrating processing modules into a pipeline. It improves over previous approaches by allowing automatic integration of new compatible modules. However, each module keeps its input/output format which must be recognized by the platform and specified by the user. Furthermore, even though the platform integrates individual processing modules, the processing chain is automatically computed due to the graph-like representation, thus making it easier for the end-user.

Current work focuses on extending the RELATE platform by integrating multiple complete annotation pipelines, as opposed to the previously described platforms which integrate individual processing modules. Furthermore, the RELATE platform makes use of a common format for data representation, similar to the CAS in the UIMA framework (as described above) while also offering direct end-user access and interaction via a web interface, without requiring any programming knowledge.

This paper is structured as follows: first, the issues related to having multiple pipelines in the platform are presented (in Section 2), followed by the resulting platform

architecture (in Section 3), then the experience of integrating UDPipe[12] (Straka and Straková, 2017) (in Section 4) and finally the conclusions.

2. Issues related to integration of multiple annotation pipelines

The RELATE platform was initially constructed around TEPROLIN's modules as the single annotation pipeline. It was further enhanced by adding new modules (such as EuroVoc, IATE, Statistics computation) at the end of the pipeline, as well as by creating links to existing resources (CoRoLa, RoWordNet, Word Embeddings, speech recognition, speech synthesis). This implementation worked well since every module was aware of the output format of the previous module and thus was able to consume the produced output.

When considering the integration of multiple pipelines, a key aspect to take into account is inter-module communication. This is extremely important if certain modules are to be re-used amongst different pipelines. A clear choice for this is constituted by modules developed outside of TEPROLIN. However, even though TEPROLIN is monolithic in nature, a few of its components (such as named entity recognition) are already implemented as web services and thus are also good candidates for reuse amongst pipelines. There are two possible approaches:

a) Implement input converters for each module to be able to understand the specific output format produced by previous modules in the pipeline. This is a complex approach, potentially requiring NxM converters, where N is the number of pipelines and M is the number of reusable modules.

b) Implement input/output converters from/to a common format. This implies constructing N output converters and M input converters. This results in a number of N+M total converters, smaller than the previous case.

Further analysis of the presented choices combined with the formats already used by some modules led to choosing CoNLL-U Plus as the "internal" format of the platform. This format is token oriented (each line represents a token with the associated annotations). Other information, such as document-level or sentence-level metadata can be included using special lines starting with "#". The first metadata line in the file acts as a simple schema by describing the contents of each column. This format choice implies converting the JSON output format of TEPROLIN to CoNLL-U as well as writing additional output converters for any new pipelines that are to be integrated. Furthermore, currently available additional annotation modules work either directly with CoNLL-U Plus or with the simpler version CoNLL-U, thus converters are not needed in this case.

Considering the space required to store uncompressed CoNLL-U Plus files, for each token a tab character is added between annotations, and a new line character is inserted at the end of the annotations list. Thus, considering N annotations the additional space required for each token due to the format used is exactly N bytes. Comparing this with JSON, for each annotation at least 4 additional characters (colon, opening and closing

[12] http://ufal.mff.cuni.cz/udpipe

quotes, comma) are needed. Considering a numeric representation of the annotation, the resulting JSON format is similar to this example: *0:"actual_annotation"*. However, the output encoding in TEPROLIN makes use of literals for the annotations which requires quotes also around the labels, thus increasing the size to a total of six additional characters. Furthermore, the JSON format specifies separating multiple list entries by commas, adding an additional character. In total, the minimum additional size introduced by JSON format while considering N annotations is 5xN-1 bytes. The XML format requires even more additional characters since it needs a token tag as well as opening and closing of the tags. Therefore, the CoNLL-U Plus format seems to make sense as an internal format also from the point of view of required additional space for storage. Even though the hierarchical nature of the JSON and XML formats may offer benefits for the end-user when annotating sequences of tokens, this can also happen in the CoNLL-U Plus format by using term identifiers, as implemented in the case of the IATE and EuroVoc annotators and integrated in the MARCELL legislative corpus (Varadi *et al*., 2020). Furthermore, JSON or XML format for a specific file may be extracted from the platform by using format converters.

Another aspect related to the integration of different pipelines is represented by the high-performance architecture of RELATE which allows demanding annotation jobs to be parallelized. This implies that new pipelines should either follow this principle (accept multiple requests and process them in parallel) or the parallelization is part of the integration into the RELATE platform. In the second case, as part of the integration, multiple processes will be created and the platform must distribute the requests amongst those processes. Annotation results are then retrieved and stored in the common internal format.

3. Platform architecture

Given the considerations presented in Section 2 regarding the common internal format, as well as parallelization of annotation jobs, the platform architecture was updated as presented in Figure 1. There are two separate stores, one for raw text files and one for annotated files in the internal format, CoNLL-U Plus. Each monolithic pipeline can interact with the collection of raw texts in order to retrieve the document to be annotated and then is allowed to store an annotated document by using a converter. Additional modules make direct use of annotated documents in the internal format (by means of input converters if needed, but these are not presented in the diagram) and finally store the output, with the new annotations included, in the internal store.

A platform user has access to both the raw text and annotated documents in the internal format. Therefore, it is possible to upload and download both raw and annotated texts. The option to upload annotated text was added to allow a user to use a pipeline which is not already included and then still use RELATE for adding additional annotations.

Finally, the platform allows annotated documents to be exported in either the internal CoNLL-U Plus format or in other formats (such as XML) by using the "Final Format Converters" as depicted in the diagram. Resulting files can then be compressed as zip archives.

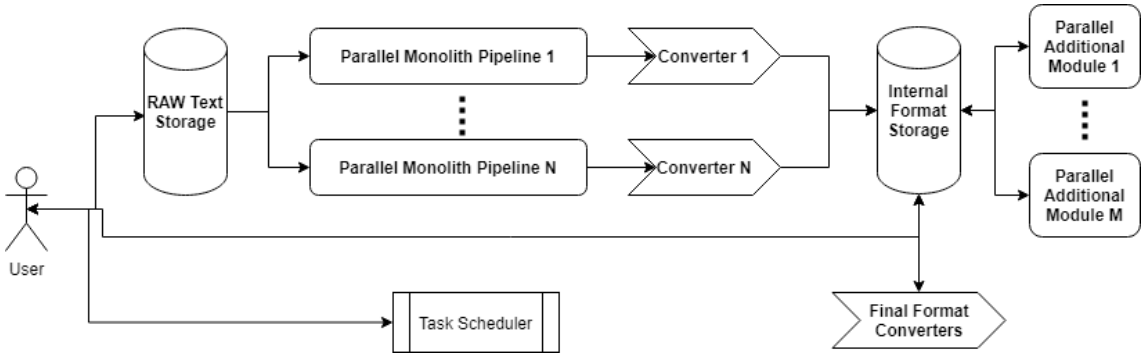


Figure 1: Platform architecture with multiple annotation pipelines

The actual annotation process is controlled by a task scheduler. The user can start an annotation process by registering a new task. Then the task scheduler starts distributing the documents amongst the different processes of a monolithic pipeline and/or the additional modules. The actual interactions between the task scheduler and the different components are not represented in Figure 1 to reduce the diagram complexity and make it easier to follow by the reader. As described by Păiș *et al*. (2020), the task scheduler is in charge of monitoring the annotation process and restarting a document annotation in case of a failure (either a process failure or a server failure). The resulting annotated documents will be stored internally in a dedicated file without combining the results from the pipelines. This is currently not an intended development for the platform.

From an implementation perspective, each module or pipeline is implemented in a dedicated folder with a JSON descriptor. This descriptor allows specifying of both GUI elements (task buttons, dialogues) and implementation files with the corresponding entry-points for scheduling and running of the annotation task. Each entry-point function will receive at runtime the platform configuration as well as input data and output destination. The module implementation is isolated from other platform components and will be invoked by the task scheduler as needed, in parallel for different input documents. The module itself is responsible for applying its own annotations and may use platform library functions for format conversions when needed.

4. Integration of UDPipe as an alternate pipeline

UDPipe is a trainable pipeline for tokenization, tagging, lemmatization and dependency parsing. It is language-agnostic and can be trained given annotated data on any language. Pre-trained models based on Universal Dependencies[13] data are available for more than 60 languages, including Romanian. This made UDPipe a good candidate for integration into RELATE as another annotation pipeline. Additionally, during the CoNLL 2018 shared task on multilingual parsing from raw text to Universal Dependencies (Zeman *et al*., 2018), NLP-Cube (on which the current TEPROLIN pipeline integrated in RELATE is based) was compared against UDPipe. According to the official ranking of the participating systems, a baseline UDPipe 1.2 system obtained

[13] https://universaldependencies.org/

a LAS F1 score of 65.8, while NLP-Cube obtained 70.82. However, a prototype version called UDPipe-Future (Straka, 2018) obtained an even higher score of 73.11.

UDPipe comes with different integration options: a command line executable, bindings to different programming languages, a REST API server and a Docker[14] container exposing the API server. After analysing the different possibilities, it seemed the easiest way to integrate UDPipe is via the use of the REST API within the Docker container. This also has the advantage of parallelization through instantiating multiple containers and distributing the requests appropriately.

The output of a UDPipe annotation request is in CoNLL-U format. According to the documentation, this is actually a valid CoNLL-U Plus format. However, in order to be fully compliant, an additional first comment line is needed in order to describe the columns present in the document. This makes the development of an output converter to the internal format chosen for RELATE to be much simpler than the converter needed for the JSON format of TEPROLIN.

Alves *et al*. (2020), in their study covering language tools for fifteen EU-official languages, including Romanian, indicate a mean processing speed of 5.6 tokens/second for NLP-Cube compared to 381.1 tokens/second for UDPipe. This is a speedup of more than 60 times, thus indicating that a similar time may be obtained by a single UDPipe process compared to the parallel NLP-Cube based pipeline previously available in RELATE. Nevertheless, parallelization by using multiple UDPipe instances is available as described above.

The resulting internal architecture of the RELATE platform is presented in Fig. 2. It follows the general architecture presented in Figure 1 and detailed in Section 3. Some elements were not included in the image for clarity (user interaction, task scheduler).

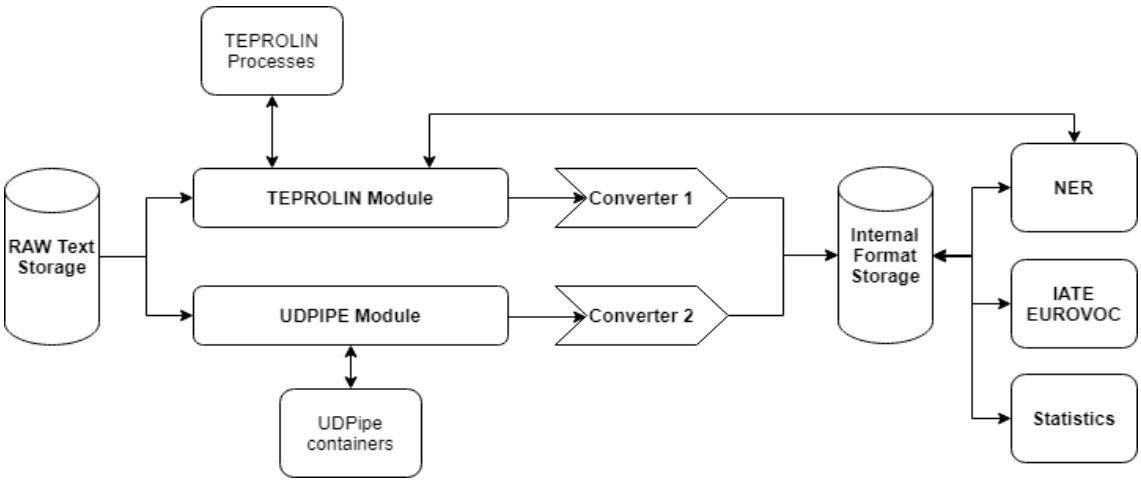


Figure 2: Internal architecture after integration of UDPipe

Following the internal integration, the UDPipe annotation pipeline was exposed in the RELATE web interface by means of a button and a task addition dialogue, as depicted in Figure 3.

[14] https://www.docker.com/

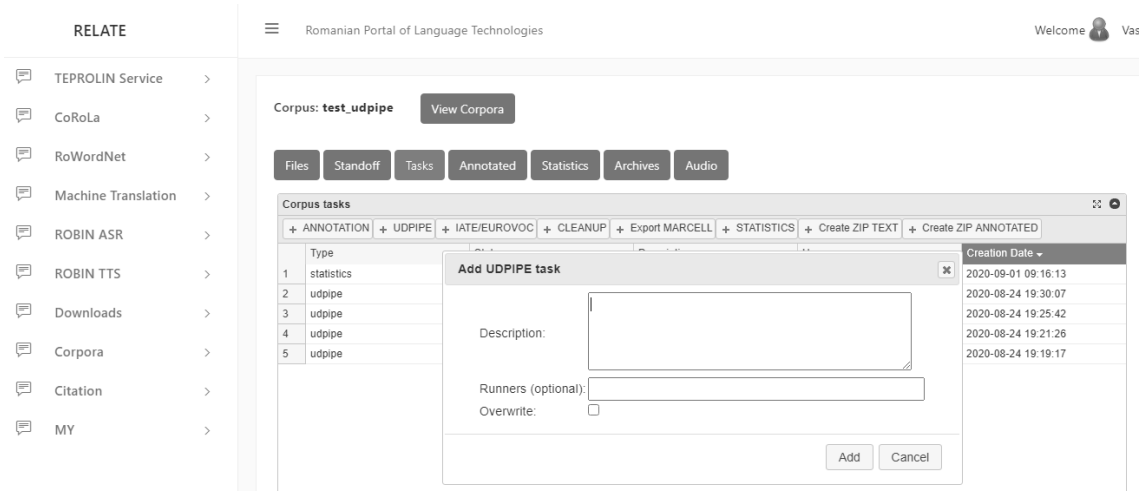


Figure 3. Interface elements allowing UDPipe annotation task to be executed

Since the result of annotation with UDPipe is finally stored in the same format used by the other platform components, no other changes in the web interface were needed. All the visualizations and interactions work similarly to the previous pipeline. In Figure 4 we present an example from a document first annotated with UDPipe and then with IATE and EuroVoc identifiers. It follows the CoNLL-U Plus format containing metadata (starting with "#", the first line describing also the file structure) and annotated tokens with the first 10 columns corresponding to basic annotations (token id, word form, lemma, part-of-speech, dependencies) and the last two columns containing EuroVoc and IATE terms (each term starts with a term number followed by a colon and the actual terminology identifier, thus allowing sequences of tokens to be annotated as a term).

```
# global.columns = ID FORM LEMMA UPOS XPOS FEATS HEAD DEPREL DEPS MISC RELATE:IATE RELATE:EUROVOC
# newdoc
# newpar
# eurovoc_domains = 04  08      10      24
# sent_id = 1
# text = - Legii nr. 500/2002 privind finanțele publice;
1       -       -       PUNCT   DASH    _       2       punct   _       _       _       _
2       Legii   lege    NOUN    Ncmpry  Case=Acc,Nom|Definite=Def|Gender=Masc|Number=Plur       0
root    _       _       50:3522916      50:1206,7206,7231
3       nr.     nr.     NOUN    Yn      Abbr=Yes        2       nmod    _       _       _       _
4       500/2002        500/2002        NUM     Mc-p-d  Number=Plur|NumForm=Digit|NumType=Card  3
nummod  _       _       _
5       privind privind ADP     Spsa    AdpType=Prep|Case=Acc   6       case    _       _       _
_
6       finanțele       finanță NOUN    Ncfpry  Case=Acc,Nom|Definite=Def|Gender=Fem|Number=Plur
3       nmod    _       _       51:1104564      51:2436,52
7       publice public  ADJ     Afpfp-n Definite=Ind|Degree=Pos|Gender=Fem|Number=Plur  6
amod    _       SpaceAfter=No   51:1104564      51:2436,52
8       ;       ;       PUNCT   SCOLON  AdpType=Prep    2       punct   _       SpacesAfter=\n  _
```

Figure 4: Part of a document annotated first with UDPipe and then with IATE and EuroVoc identifiers

5. Conclusions

The use of a single internal format, in the form of CoNLL-U Plus, as well as the architecture principles presented in Section 3 allow the RELATE platform to use different pipelines for the annotation of massive volumes of texts. This was demonstrated by the successful integration of UDPipe as a second annotation pipeline

within the RELATE platform, as described in Section 4. Furthermore, existing modules which are not part of a monolithic pipeline can be reused to provide additional annotations regardless of the pipeline being used.

Even though RELATE was initially constructed as a platform for processing Romanian language, the successful integration of UDPipe with pre-trained models, opens up the possibility to annotate multilingual corpora, containing documents written in other languages than Romanian. The same internal format is valid for other languages and currently from the available additional modules at least the "statistics" module can be reused regardless of the language.

The RELATE platform is open source and available in a GitHub repository[15]. This includes all the components, including integrations presented in this paper. A current running version of the platform is in use at RACAI, on its own server[16].

Acknowledgements

Part of this work was conducted in the context of the ReTeRom project. Another part of this research was supported by the EC grant no. INEA/CEF/ICT/A2017/1565710 for the Action no. 2017-EU-IA-0136 entitled "Multilingual Resources for CEF.AT in the legal domain" (MARCELL). And a last part of this research was supported by the EC grant no. INEA/CEF/ICT/A2019/1926831 for the Action no. 2019-EU-IA-0034 entitled "Curated Multilingual Language Resources for CEF AT" (CURLICAT).

References

Alves, D., Thakkar, G., and Tadić, M. (2020). Evaluating Language Tools for Fifteen EU-official Under-resourced Languages. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, ELRA, 1866-1873.

Avram, A.M., Păiș, V., Tufiș, D. (2020). Towards a Romanian end-to-end automatic speech recognition based on DeepSpeech2. *Proceedings of the Romanian Academy, Series A*, in-print.

Bański, P., Fischer, P.M., Frick, E., Ketzan, E., Kupietz, M., Schonefeld, O., and Witt, A. (2012). The New IDS Corpus Analysis Platform: Challenges and Prospects. In N. Calzolari, K. Choukri, T. Declerck, M. Uğur Doğan, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, S. Piperidis (eds), *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC)*, Istanbul, Turkey, 2905-2911.

Boroș, T., Dumitrescu, S.D., and Burtică, R. (2018a). NLP-Cube: End-to-end raw text processing with neural networks. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, ACL, 171-179.

Boroș, T., Dumitrescu, S.D., and Păiș, V. (2018b). Tools and resources for Romanian text-to-speech and speech-to-text applications. In *Proceedings of the International Conference on Human-Computer Interaction - RoCHI*, 46-53.

[15] https://github.com/racai-ai/RELATE
[16] http://relate.racai.ro

Coman, A., Mitrofan M., and Tufiș, D. (2019). Automatic identification and classification of legal terms in Romanian law texts. In *Proceedings of the International Conference on Linguistic Resources and Tools for Processing Romanian Language (ConsILR)*, 39-49.

Cristea, D., and Butnariu, C. (2004). Hierarchical XML representation for heavily annotated corpora. In *Proceedings of the LREC 2004 Workshop on XML-Based Richly Annotated Corpora*, Lisbon, Portugal.

Cristea, D., and Pistol, I.C. (2008). Managing Language Resources and Tools Using a Hierarchy of Annotation Schemas. In *Proceedings of the Workshop on Sustainability of Language Resources*, LREC-2008, Marrakech.

Cunningham, H. (2002). GATE, A General Architecture for Text Engineering. *Computers and the Humanities*, 36(2), 223-254.

Ferrucci, D., Lally, A. (2004). UIMA: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering 10*, 3-4, 327-348.

Ion, R. (2007). Word Sense Disambiguation Methods Applied to English and Romanian. *PhD Dissertation*, Romanian Academy.

Ion, R. (2018). TEPROLIN: An Extensible, Online Text Preprocessing Platform for Romanian. In *Proceedings of the International Conference on Linguistic Resources and Tools for Processing Romanian Language (ConsILR)*, Iași, Romania, 69-76.

Ion, R., Badea, V.G., Cioroiu, G., Barbu Mititelu, V., Irimia, E., Mitrofan, M., Tufiș, D. (2020). A Dialog Manager for Micro-Worlds. *Studies in informatics and control*, 29:4, 411-420.

Păiș, V., Tufiș, D. (2018). Computing distributed representations of words using the COROLA corpus. In *Proceedings of the Romanian Academy, Series A*, Volume 19, 2, 403-409.

Păiș, V., Tufiș, D., and Ion, R. (2019). Integration of Romanian NLP tools into the RELATE platform. In *Proceedings of the International Conference on Linguistic Resources and Tools for Processing Romanian Language (ConsILR)*, 181-192.

Păiș, V., Tufiș, D., and Ion, R. (2020). A Processing Platform Relating Data and Tools for Romanian Language. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, ELRA, Marseille, France, 81-88.

Perovšek, M., Kranjc, J., Erjavec, T., Cestnik, B., Lavrač, N. (2016). TextFlows: A visual programming platform for text mining and natural language processing. *Science of Computer Programming*, 121, 128-152.

Pistol, I.C. (2011). The Automated Processing of Natural Language. *Ph.D. Dissertation*, "A.I. Cuza" University of Iași.

Straka, M. (2018). UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Association for Computational Linguistics, 197-207

Straka, M., Straková, J. (2017). Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Vancouver, Canada, 88-99.

Tufiș, D., Barbu Mititelu, V. (2014). The Lexical Ontology for Romanian. In N. Gala, R. Rapp, N. Bel-Enguix (eds), *Language Production, Cognition, and the Lexicon*, series Text, Speech and Language Technology, vol. 48, Springer, 491-504.

Tufiș, D., Barbu Mititelu, V., Irimia, E., Păiș, V., Ion, R., Diewald, N., Mitrofan, M., Onofrei, M. (2019). Little strokes fell great oaks. Creating CoRoLa, the reference corpus of contemporary Romanian. *Revue Roumaine de linguistique*, LXIV (3), 227-240.

Tufiș, D., Mitrofan, M., Păiș, V., Ion, R., and Coman, A. (2020) Collection and Annotation of the Romanian Legal Corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference*. ELRA, Marseille, France, 2766-2770.

Váradi, T., Koeva, S., Yamalov, M., Tadić, M., Sass, B., Nitoń, B., Ogrodniczuk, M., Pęzik, P., Barbu Mititelu, V., Ion, R., Irimia, E., Mitrofan, M., Păiș, V., Tufiș, D., Garabík, R., Krek, S., Repar, A., Rihtar, M., and Brank, J. (2020). The MARCELL Legislative Corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*. ELRA, Marseille, France, 3754-3761.

Zeman, D., Hajič, J., Popel, M., Potthast, M., Straka, M., Ginter, F., Nivre, J., and Petrov, S. (2018). CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, ACL, 1-21.

EXPLORING VARIATIONAL AUTOENCODERS FOR LEMMATIZATION

PETRU REBEJA

Faculty of Computer Science, Alexandru Ioan Cuza University of Iași,

petru.rebeja@gmail.com

Abstract

In this paper we present two experiments in which we attempt to train Variational Autoencoders on word embeddings: one for the common use-case of generating the word embeddings similar to the ones used for training, and the second - to generate the embeddings of the lemmas of words used for training. In this way we try to approach the task of lemmatization using Variational Autoencoders.

Key words — CoRoLa Word Embeddings, Lemmatization, Variational Autoencoder.

1. Introduction

Lemmatization is the process of determining the base form (lemma) of an inflected word. This process bears special significance in Natural Language Processing for morphologically-rich languages because it offers support for various other downstream tasks such as text-based search, word clustering etc.

In this paper we present the experiments set up to investigate whether the task of lemmatization can be tackled using the Variational Autoencoder models (Kingma and Welling, 2014; Rezende *et al*., 2014). In order to do so, we create a training set by gathering data from multiple corpora and building a list of word embeddings and lemma embeddings which we use to train the model.

This paper is structured as follows: Section 2 presents a high-level description of Variational Autoencoders, Section 3 explains why we choose to conduct this experiment and briefly mentions related work, Section 4 describes the data used for training the model and how the data was gathered, Section 5 describes the experiments and the results, and in the end, Section 6 presents the conclusions and future work.

2. Variational Autoencoders

Variational Autoencoders (Kingma and Welling, 2014; Rezende *et al.*, 2014) are a special type of probabilistic deep-learning models that have proven to be successful in generating images similar to the images used in training (Gregor *et al.*, 2015), *i.e.*, the model learns to generate images that are closely similar to a base image received as the input.

A Variational Autoencoder (VAE) consists of two main components, namely an encoder and a decoder. The encoder is a neural network that learns a sufficiently complicated function which maps the input of the network to a space of latent variables

(Doersch, 2016), and the decoder is another neural network that is able to reconstruct (decode) a data sample from its latent representation. The components of the VAE optimize the negative of the *Evidence Lower BOund* (ELBO) function which, in its turn has two components: (i) how well the decoder is able to reconstruct the data sample, and (ii) how the distribution of the latent variables models the distribution of training data, measured through the Kullback-Leibler divergence.

3. Motivation and related work

Since, as mentioned in Section 2, Variational Autoencoders are able to generate images that are similar to the ones used in training, we can posit that the model is able to capture the "essence" of an image into its latent variables and generate the new ones by applying some transformations to the "non-essential" parts. If we were to transfer this process to the domain of Computational Linguistics, specifically, to the lemmatization task where the "essence" of a word is its lemma, then the inflection process would be akin to generating words "similar" to the lemma. As such, if the Variational Autoencoder is able to capture a representation of the lemma into the latent space, then the decoder network should be able to reconstruct the lemma or, at least construct a point in the hyperspace such that it is close to the lemma.

Our main objective is to determine the lemma of a given word; thus, we want to see how good the VAE model is at reconstructing the lemma from a given word. However, we need to ensure first that the VAE model is applicable in such situation, thus we also want to see whether the model is able to perform its original task – generating samples that are similar to the ones received as input – in the domain of Computational Linguistics.

To the best of our knowledge, there are no other attempts to tackle the lemmatization task for the Romanian language using Variational Autoencoders. Another approach to lemmatization is presented by Boros *et al.* (2018) who introduced NLP-Cube. Unlike NLP-Cube, our approach uses a different model and is trained on a completely separate set of data.

4. Data and preprocessing

In order to train a Variational Autoencoder to generate the lemma of a given inflected word, the training data needs to be subjected to two constraints: (i) each training sample should consist of an inflected word given as input and its associated lemma from which the model will calculate the reconstruction loss, and (ii) both the inflected words and its associated lemma must be provided as a numerical form.

Therefore, in building our dataset we faced two problems: (i) finding a sufficiently large association set of inflected words and their lemmas, and respectively (ii) determining an encoding scheme for word-lemma pairs.

Given the wide popularity and success of word embeddings (Mikolov *et al.*, 2013) in deep learning research we decided to stay on the beaten path and use the pretrained

word-embeddings for Romanian language from the CoRoLa corpus (Barbu Mititelu *et al.*, 2014), which are available online[1].

For the other challenge, namely the collection of word-lemma pairs used for training, we used the MARCELL corpus (Tufis *et al.*, 2020) as the source of our data. The MARCELL corpus consists of a large collection of legal documents annotated in XML format and the annotation schema provides both the inflected word as it was used in the source document and its lemma.

We refrained from loading the entire annotated document and extracting the word-lemma pairs on the fly as it would have put a high pressure on memory usage for the entire system. Instead, we created a Python script to process the MARCELL corpus offline. The script loads the files from the corpus one by one, extracts the inflected words and their lemmas from the document, and saves them into a global collection of unique pairs. After processing all the documents from the corpus, the list of unique word-lemma pairs is saved into a file on disk which will serve later as the input of the training pipeline.

However, after inspecting the resulting file, we found out that out of the 783,568 word-lemma pairs, many of them contained non-alphabetic characters and punctuation marks in either the inflected word, lemma, or both. To exclude such entries, we changed the extraction script so as to discard the pairs that contain any character other than letters, dash (-), and apostrophe ('), which reduced the number of pairs to 359,688.

Upon further inspection we found that the reduced list contains invalid words which also needed to be filtered-out by using a dictionary. In order to build a list of valid words forms we downloaded a database dump of the dexonline application[2] which contains the list of copyright-free dictionary entries[3] from which we extracted the list of word inflections into a separate text file.

The files containing word-lemma pairs, valid word forms, word embeddings, and lemma embeddings are processed by yet another `Python` script that performs an in-memory intersection of these data sets resulting in 93,190 records of the form *(w, we, l, le)* where: *w* is the inflected word, *we* is the word embedding, *l* is the lemma of *w*, and *le* is the lemma embedding.

5. Experiments and results

After building and validating the default architecture of a Variational Autoencoder, we tweaked our model to accept one more parameter, namely the reconstruction target. This parameter is meant to tell the model what to reconstruct from the word-lemma pair, thus allowing us to conduct two different experiments on the same architecture:

- how well the model is able to reconstruct the word embedding given as input; this experiment is similar to previous usages of VAE models, the only difference being that instead of reconstructing an image, the model is reconstructing a word embedding, and

[1] http://corola.racai.ro/
[2] https://dexonline.ro/
[3] https://wiki.dexonline.ro/wiki/Informații

- how well the model is able to satisfy our target objective, namely to reconstruct the lemma embedding for a given word embedding.

The reconstruction target parameter (`reconstruct_lemma`) is passed in the constructor of the VAE model class and is used only when training the model to determine which part of the input data is used to calculate the loss. At each training step the model splits the input pair of word and lemma embeddings into its constituent parts and based on the value of the `reconstruct_lemma` parameter decides which part will be used to calculate the reconstruction loss as shown in the pseudo-code below:

```
word, lemma = data[0], data[1]
target = lemma if reconstruct_lemma else word
z_mean, z_log_var, z = encoder(word)
reconstruction = decoder(z)
reconstruction_loss= calculate_reconstruction_loss(reconstruction,target)
kl_loss = calculate_kl_loss(z_mean, z_log_var)
total_loss = reconstruction_loss + kl_loss
```

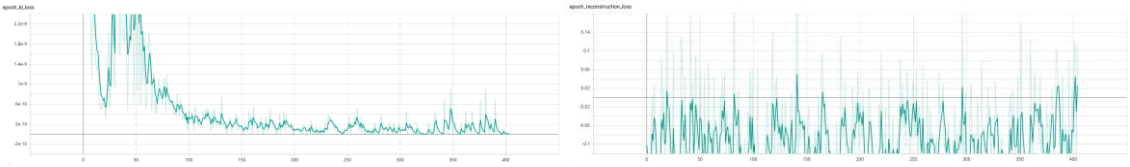
We trained the model in both our experiments with the Adam (Kingma and Ba, 2014) optimizer, the hyper-parameters presented in **Table 1**, and the binary-crossentropy loss function which is equivalent to minimizing the negative logarithm of a Bernoulli distribution. The evolution of the loss value for each of the above experiments is presented in **Figure 1** – for generating word embeddings, and respectively, in **Figure 2** for generating the lemma embeddings.

Table 1: The hyper-parameters used for training the Variational Autoencoder model. The Embedding dimensions parameter is not configurable and is determined from the CoRoLa embeddings.

Parameter name	**Value**
Number of latent dimensions	128
Embedding dimensions	300
Hidden dimensions	4,096
Batch size	128
Training epochs for lemma embeddings	600
Training epochs for word embeddings	400
Adam optimizer parameters	
Learning rate	1
Decay rate β_1	0.9
Decay rate β_2	999

As can be seen from both **Figure 1** and **Figure 2**, although the model does approximate the underlying distribution of words and lemmas, shown by the small Kullback-Leibler divergence value, to our great disappointment it fails to reconstruct both the word and lemma embeddings. Furthermore, although exactly the same model architecture was used for both our experiments, we see an odd behaviour for the reconstruction loss of word embeddings, namely the loss is mostly negative. Usually, this behavior is due to input data not being normalized; however, subsequent verification of the source code

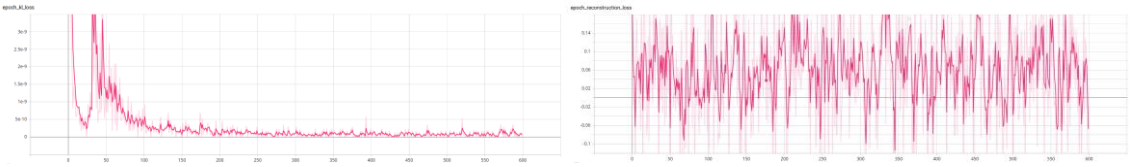
showed that the data was normalized before training. We do not know the cause of this effect, but we consider that finding and fixing the error may be the key to at least better results in our experiments.



(a) Kullback-Leibler divergence loss for generating word embedding.

(b) Reconstruction loss for generating word embedding.

Figure 1: Loss graphs for the experiment meant to generate word embeddings similar to the ones used as the input of the VAE model.



(a) Kullback-Leibler divergence loss for generating lemma embedding.

(b) Reconstruction loss for generating lemma embedding.

Figure 2: Loss graphs for the experiment meant to generate lemma embeddings of the word embeddings used as the input of the VAE model.

The behaviour mentioned above is also one of the reasons for the difference in the number of training epochs for each experiment: since there was no improvement on the reconstruction loss of lemma embedding the training, was stopped.

6. Conclusions and future work

In this paper we present two experiments meant for exploring the usage of Variational Autoencoder architectures for lemmatization. In the first experiment, we investigate the results of applying the traditional use-case of Variational Autoencoders, namely generating new data points similar with the ones used in training the model, on word embeddings. With this experiment we want to see if the VAE model will be able to generate similar word embeddings to the ones seen during training. The second experiment aims to go beyond the traditional use case and to investigate whether the VAE architecture is able to generate embeddings similar to the embeddings of the lemmas of words seen during training.

Our results show that for both experiments, the models are able to closely approximate the underlying distributions of embeddings but they fail to reconstruct the expected embeddings from the latent variables. Furthermore, we observed an unusual behaviour of the model when reconstructing the lemma that results in negative values for the loss function.

In order to improve the results, we need to investigate and fix the root cause of the behaviour mentioned above and only afterwards we can proceed with the traditional phase of fine-tuning the hyper-parameters of the model. A second approach is to change the loss function from a discrete Bernoulli likelihood to using a continuous Bernoulli distribution (Loaiza-Ganem and Cunningham, 2019).

References

Barbu Mititelu, V., Irimia, E. and Tufis, D. (2014). CoRoLa - The Reference Corpus of Contemporary Romanian Language. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, LREC 2014, Reykjavik, Iceland, 1235–1239.

Boros, T., Dumitrescu, S. D. and Burtica, R. (2018). NLP-cube: End-to-end Raw Text Processing with Neural Networks. In *Proceedings of the CoNLL 2018 Shared Task "Multilingual Parsing from Raw Text to Universal Dependencies"*, Brussels, Belgium, 171–179.

Doersch, C. (2016). Tutorial on variational autoencoders. *Computing Research Repository*, 16:6, 2331-8422.

Gregor, K., Danihelka, I., Graves, A. and Wierstra, D. (2015). DRAW: A Recurrent Neural Network for Image Generation. In *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, 1462-1471.

Kingma, D.P. and Welling, M. (2014) Auto-encoding Variational Bayes. In *2nd International Conference on Learning Representations*, Banff, AB, Canada.

Kingma, D.P. and Ba, J. (2014) Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations*, San Diego, CA, USA.

Loaiza-Ganem, G. and Cunningham, J.P. (2019). The Continuous Bernoulli: Fixing a Pervasive Error in Variational Autoencoders. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*, NeurIPS 2019, Vancouver, BC, Canada, 13266–13276.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J. (2013). Distributed Representations of Words and Phrases and Their Compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013*, Lake Tahoe, Nevada, United States, 3111–3119.

Rezende, D. J., Mohamed, S. and Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31th International Conference on Machine Learning*, ICML 2014, Beijing, China, 1278-1286.

Tufis, D., Mitrofan, M., Păiș, V., Ion, R. and Coman, A. (2020). Collection and Annotation of the Romanian Legal Corpus. In *Proceedings of The 12th Language Resources and Evaluation Conference*, Marseille, France, 2773–2777.

A WORD SENSE ALIGNMENT APPROACH BASED ON THE ROMANIAN WORDNET AND EDTRL RESOURCES

ANDREI SCUTELNICU

Faculty of Computer Science, "A. I. Cuza" University of Iași
Institute for Computer Science, Romanian Academy, Iași branch

andreis@info.uaic.ro

Abstract

The Romanian Wordnet is an impressive collection of Romanian nouns, verbs, adjectives, and adverbs, which can be seen as a network of synonymy sets, each such set being made up of words labelled by senses. In this study, we propose a model for aligning the word senses of the Romanian Wordnet with those of the Thesaurus Dictionary of the Romanian Language in electronic form, by exploiting the collections of definitions and examples in the two linguistic thesauri, based on a statistical model.

Key words — eDTLR, lexical resources, RoWordNet, statistical model, word sense

1. Introduction

The automatic process of word sense disambiguation (WSD) has been a topic of interest since the 1950s (period in which studies in the field of computational linguistics began to intensify). Sense disambiguation is not a goal in itself; it is an intermediate process, necessary at a certain level in the natural language processing chain. It is useful for applications that require language interpretation (communication through messages, human-machine interaction).

The problem of WSD has been described as AI-complete: a problem is AI-complete if it can be solved only by solving all the difficult problems in artificial intelligence (AI), such as the representation of the meanings of words and knowledge. The difficulty of WSD was one of the central points of Bar-Hillel's thesis (1960) in the field of machine translation, in which he argued that there is no possibility to automatically determine the meaning of the word "pen" in the sentence: "The box is in the pen". Bar-Hillel's argument was the basis for the ALPAC report, which is considered one of the reasons for the abandonment of most of the automatic translation study projects in the '60s.

On the other hand, at the same time, enormous progress was being made in the field of knowledge representation. The semantic networks have appeared which will be applied in the study of the sense disambiguation. In the last ten years, there has been an intensification of the efforts of automatic WSD, due to the increased access to text processed by the machine, as well as due to the improvement of the statistical methods of identifying and applying the models on the data.

In our paper, we propose a word sense alignment model based on the Romanian Wordnet (RoWN) and the Thesaurus Dictionary of the Romanian Language in

electronic form (eDTLR) resources. This is a first step that can lead to full word sense disambiguation.

2. Lexical resources

The existence of the two resources, RoWN and eDTLR, in electronic format, inevitably led to the idea of aligning them. RoWordNet[1] (Tufis and Cristea, 2002; Tufis *et al.*, 2004; Tufis *et al.*, 2013) is created according to Princeton WordNet[2] (Fellbaum, 1998) principles and can be considered a real revolution in Romanian computational linguistics, through the scientific advances that has generated in recent years. This database is an impressive collection of nouns, verbs, adjectives, and adverbs that we can see as a network of nodes, where we find words that have the same meaning in a certain context, and in the mesh of that network are found the other words that, in their turn, belong to a network node with (an)other word(s) that share their meaning in a specific context. These are grouped into sets of cognitive synonyms called synsets, each expressing a distinct concept, representing virtually the node of the network that will determine the set in which the search for a given context is made. Each has a unique synset identifier and groups a set of lexical items associated with their sense IDs; on this basis, the sense of a word searched in a context can be extracted. Nouns and verbs are organized in a conceptual hierarchy through hypernymic and / or hyponymic relationships.

eDTLR[3] - Thesaurus Dictionary of the Romanian Language in Electronic Format – (Cristea *et al.*, 2011) represents an impressive collection of words that synthesize the entire evolution of the language from origins to present. It is especially important to emphasize the need for such a resource for any language. Practically, this Thesaurus Dictionary is a veritable incursion into the written and/or spoken history of a people, as it sums up a huge amount of information about a language, in this case, the Romanian language (Haja *et al.*, 2005). Thus, within this resource, we can first identify different entries of the same word (meaning homonymous), for which we can identify meanings and subsenses. For each meaning and subsense, we can find information about the origin of a word, general and/or specific definitions, and literary examples (quoted in Romanian literature) that will help us have an overview of the significance of that meaning. Romanian is a difficult language through the complexity of interpretation, with multiple meanings, interpretable in certain contexts by the multitude of semantic senses. It is also difficult to interpret these senses in the written language, as sometimes the deduction of certain aspects is affected by the inflections of the speaker's voice; often a written quotation can obtain different resonances in the mind of the reader through the simple inflection of the voice. This is why it is necessary to identify multiple quotes from literature for one sense or subsense, to make sure we can correctly position the meaning of a word in a given context.

[1] http://www.racai.ro/en/tools/text/rowordnet/
[2] https://wordnet.princeton.edu/
[3] http://edtlr.info.uaic.ro/

3. Sense alignment algorithms

The format of the entries of the two resources is of XML type; thus, a first step for aligning them was the writing of procedures for accessing the information contained in the XML structure. Having the necessary extracted information, the next step was the development of scoring functions to measure the degree of similarity between the meanings of words. Then came the actual alignment, calculating a new score from the scores calculated by the functions from the previous step, but also taking into account other important information, such as the part of speech.

3.1. Scoring functions

Having extracted all the necessary information about words, the next step is to elaborate some heuristics that will be the basis of the alignment. Starting from the model proposed by Kwong (1996), the proposed scoring functions have as main support the co-occurrence of terms, calculating to what extent the definitions and the examples also overlap.

3.1.1 First scoring function

This function analyzes how the synonyms of the word are used in the definitions of meanings. The prototype of the function is:

```
double [] firstFunction (Vector syns, Vector defs)
```

The two Vector-type objects transmitted as parameters represent the synonyms of the word (obtained from RoWN sysnset), and the definitions of meanings respectively (obtained from eDTLR). The function returns a table of values representing the score obtained in each definition. The calculation of the score is done by a simple procedure, practically counting the occurrences of synonyms in definitions. Thus the score obtained by each definition is actually the number of synonyms present within it.

3.1.2. Second scoring function

This function analyses how the synonyms of the word are found in the examples of meanings. The prototype of the function is:

```
double secondFunction (Vector syns, Vector examples)
```

The two Vector-type objects transmitted as parameters represent the word synonyms obtained from RoWN synsets, and the examples of a sense from eDTLR, respectively. The function returns a value that represents the score obtained by the meaning for which the examples were extracted. The calculation of the score is done in the same way as for the previous function: for each example, count how many of the synonyms are found in it and then add this number to the total score. Thus, the score obtained is actually the sum of the number of synonyms present in each example.

3.1.3. Third scoring function

This function analyses how the definition of the synset in which the word is found overlaps with the definitions of the eDTLR meanings of the word. The prototype of the function is:

```
double [] thirdFunction (String syn_def, Vector defs)
```

The two arguments transmitted as parameters represent the definition of the synset - parameter of type String - obtained from RoWN and definitions of meanings - parameter of type Vector - obtained from eDTLR. The function returns a table of values representing the score obtained by each definition. The vector has the same size as the definition vector. The calculation of the score is done in several steps:

- divide the definition of the synset into lexical units;
- count the lexical units, obtained in the previous step, within each meaning definition in eDTLR; thus, a preliminary score, *s1*, is obtained for each definition;
- divide each definition from the definition vector received as an argument into lexical units and count them; thus, for each definition, a number *n* is obtained;
- obtain the final score, *s*, by dividing the preliminary score of the definition by its number of lexical units: $s = s1 / n$

Thus, the score obtained by each definition is in fact the ratio between the number of lexical units in the definition of the synset present in the definition of the meaning in eDTLR for which the score is calculated and the number of lexical units of the same definition.

3.1.4. Fourth scoring function

This function analyzes how the definition of the synset in which the word is found overlaps with the examples of a sense in the word eDTLR. The prototype of the function is:

```
double forthFunction (String syn_def, Vector examples)
```

The two arguments transmitted as parameters represent the definition of the synset - the String type parameter - obtained from RoWN and the examples of a sense - Vector type parameter - obtained from eDTLR. The function returns a single value that represents the score obtained by the meaning for which the examples were extracted. The calculation of the score is done, to a large extent, as in the previous function:

- divide the definition of the synset into lexical units;
- count the lexical units obtained in the previous step, within each example of meaning; then the number obtained is added to the total sense score; thus, we obtain a preliminary score for meaning, *s1*;
- compute the final score, s, by dividing the preliminary meaning score into the number of examples of the meaning in the searched case: $s = s1 / nr\_example$

Thus, the score obtained by the intended meaning is in fact the ratio between the sum of the lexical unit numbers in the definition of the synset present in each example of it and the number of examples of the meaning from eDTLR.

3.2. Word Sense Alignment

With the possibility of calculating scores for the meanings of a word, the last step is to combine these scores to determine the meaning closest to the meaning of RoWN. In the first phase, however, a selection of the synsets is made. This selection is based on the fact that the input word must have the same part of speech in both resources. After this selection we can move further with the effective calculation of the final score and assign the winner.

3.2.1. Sorting function

This function removes from the list of synsets found for a word those synsets that do not have the same part of speech as the word in the dictionary. The analysis mechanism must take into account that the parts of speech are not represented equally in these two used resources. Thus, it must retain an alignment of the specifiers for the part of speech. So far, the part of speech alignment is done explicitly, using conditions such as: if the part of speech of the synset and the part of speech of the dictionary entry are the same, then keep; if not, delete.

3.2.2. Alignment mechanism

The alignment function receives as a parameter an object of type String, which represents the word for which the alignment of the senses is desired.

The first step is the population of the synset vectors, respectively occurrences for the received word. This step is followed by testing the existence of the desired word in the two resources: if it does not exist in any of the resources, then the application displays this and quits the alignment process. Then, the number of senses for that word is verified; if it appears with only one meaning in both resources, then the application no longer goes into alignment, because it is no longer meaningful, and signals that a “one-to-one” type alignment is attempted.

The next step is to match the parts of speech. In this step, the function for sorting the synsets is used, keeping within the sysnset vector only those with the corresponding part of speech. Then, the scoring functions are applied, calculating a score for each combination of synset (from RoWN) and meaning (from eDTLR). Thus, each score calculated with the four calculation functions is kept within a two-dimensional matrix, where the first dimension is the number of synsets and the second is the number of senses. The final score is the sum of the previously obtained scores. One can also try a weighting of these scores; this variant was not taken into account because some scores were weighted by the calculation mechanism itself.

Also, the alignment is made by taking the maximum element for each line: thus, the synset *i* has the corresponding *j* meaning of a word if the element *i,j* is the maximum on line *i*, corresponding to the synset *i*.

4. Results and conclusions

To test the proposed alignment mechanism, the latest version of RoWN (Barbu Mititelu *et al.*, 2014) and the volume containing the letter V (from the word "*venial*" to the word "*vizurina*") from eDTLR have been used.

These two resources share only 610 entries, including one-way entries. Approximately 591 words in RoWN that are found, in lexicographic order, between "*venial*" and "*vizurina*" are not found in eDTLR, while approximately 4700 words in eDTLR are not found in RoWN.

For a series of common entries found, the alignment mechanism was tested, but a correct evaluation and the formulation of a clear conclusion will be possible only when a manual alignment of the meanings for some entries will be made. Manual alignment will reveal whether it overlaps with the one provided by the algorithm. Following the alignments provided by the algorithm, however, we can say that a larger-scale approach is worth trying.

Another approach of word sense alignment which we will consider is using a neural network back-propagation algorithm, where the training will be based on the RoWN entries and the tests will be done on eDTLR; manual alignment is needed also in this case, in order to be able to compare the results and give a *trusted score* to the algorithms.

References

Bar-Hillel, Y. (1960). The present status of automatic translation of languages. In *Advances in computers*, Vol. 1, Elsevier, 91-163.

Barbu Mititelu, V., Dumitrescu, S.D. and Tufis, D. (2014). News about the Romanian WordNet. In *Proceedings of the 7th Global Wordnet Conference*, GWC 2014, 268-275.

Cristea, D., Haja, G., Moruz, A., Răschip, M. and Patrașcu, M.I. (2011). Partial statistics at the end of the eDTLR project - Thesaurus Dictionary of the Romanian Language in electronic format. In Rodica Zafiu, Camelia Ușurelu, Helga Bogdan Oprea (editors), Romanian language. Hypostases of linguistic variation. In *Romanian language. Hypostasis of linguistic variation. Acts of the 10th Colloquium of the Chair of Romanian Language*, 213-224.

Fellbaum, C. (ed.) (1998). *WordNet: An Electronic Lexical Database*, Cambridge, MA, Mit Press, 1998.

Haja, G., Danila, E., Forascu, C. and Aldea, B.M. (2005). *Dictionary of the Romanian language (DLR) in electronic format. Acquisition studies*. Alfa publishing house, Iasi, 2005.

Tufis, D. and Cristea, D. (2002). Methodological issues in building the Romanian Wordnet and consistency checks in Balkanet. In *Proceedings of the Workshop on Wordnet Structures and Standardization, and how these affect Wordnet Applications and Evaluation*, in conjunction with LREC-2002, Las Palmas, Spain.

Tufis, D., Barbu, E., Barbu Mititelu, V., Ion, R. and Bozianu, L. (2004). The Romanian wordnet. *Romanian Journal on Information Science and Technology*, Dan Tufiş (ed.) Special Issue on BalkaNet, Romanian Academy, 7(2-3), 105-122.

Tufiș, D., Barbu, E., Barbu Mititelu, E., Stefanescu, D. and Ion, R. (2013). The Romanian Wordnet in a nutshell. *Language resources and evaluation*, 47(4), 1305-1314.

CHAPTER 3.
SPEECH RECOGNITION AND SYNTHESIS

EXPLORING END-TO-END NEURAL TEXT-TO-SPEECH SYNTHESIS FOR ROMANIAN

MARIUS DUMITRACHE, TRAIAN REBEDEA

Faculty of Automatic Control and Computers, Computer Science Department, University Politehnica of Bucharest

marius.mdumitrache@gmail.com, traian.rebedea@upb.ro

Abstract

Traditional TTS systems require extensive domain expertise to adapt to new languages and datasets. Novel deep learning approaches greatly reduce the need of extensive domain expertise by replacing the traditional TTS pipeline with a neural network architecture. TTS is an especially difficult task for Romanian because of speech datasets scarcity. To our knowledge, there are very few TTS systems for Romanian and even fewer that exhibit close to natural speech behaviour. We aim to improve Romanian TTS domain by experimenting with two neural architectures: Tacotron 2 and WaveRNN. We explore three Romanian speech datasets: Romanian Speech Synthesis (RSS), Romanian Read Speech Corpus (RSC), and SWARA Speech Corpus. RSC trained models provide the least natural audio samples, whereas RSS trained models generate the most natural samples. To evaluate our results, we conduct a mean opinion score (MOS) where subjects are asked to rate the naturalness of audio samples with a score from 1 to 5. We show that our choice of neural network achieves a MOS of 4.197, whereas the audio samples generated from the current best available Romanian TTS system achieved a MOS of 3.732.

Key words — deep learning, neural networks, Romanian TTS, Tacotron, text to speech.

1. Introduction

Text to Speech (TTS) or speech synthesis is a topic at the border of signal processing and natural language processing, aiming to generate human speech. Designing a traditional TTS pipeline is laborious and needs extensive domain expertise, making it very difficult to adapt to new languages and datasets (Taylor, 2009). Romanian speech datasets scarcity makes it an especially difficult task to build a Romanian TTS system. To our knowledge, the only Romanian TTS systems that produce good quality speech are the speech synthesis systems using Hidden Markov Models (HMM) and Deep Neural Networks (DNN) proposed by Stan *et al.* (2011) and Stan *et al.* (2017). Other TTS systems of which we are aware are Ivona, a TTS system based on unit selection (Hunt and Black, 1996), and a commercially TTS system proposed by Microsoft[1].

With the advances in deep learning, speech synthesis has seen an uprising development in the recent period, especially with the development of WaveNet (Oord *et al.*, 2016), which is a neural network vocoder that can synthesize speech with human-like naturalness. Despite its results, the WaveNet architecture requires a large number of

[1] https://azure.microsoft.com/en-us/services/cognitive-services/text-to-speech (Last accessed 10.11.2020)

layers to be computed for each sample, allowing it to generate audio samples at only 0.3 times real time. To overcome this problem, WaveRNN introduced a simplified neural network vocoder architecture made of a single-layer recurrent neural network with a dual softmax layer that is on par with WaveNet in terms of speech quality, but with a generation speed of up to 4 times faster than real time (Kalchbrenner *et al.*, 2018). WaveNet and WaveRNN neural vocoders generate speech from mel-spectrograms which are low-level acoustic representations. Mel-spectrograms can be predicted by Tacotron 2 neural network or generated from ground truth audio samples. Tacotron 2 (Shen *et al.*, 2017) is a neural network with an encoder and a decoder architecture that predicts mel-spectrograms from input text. To complete the TTS pipeline, the mel-spectrograms can be transformed to audio samples with the Griffin-Lim (Griffin and Lim, 1984) algorithm or with any of the WaveNet and WaveRNN neural networks. Griffin-Lim algorithm is a fast and cheap vocoder variant to use, but the generated audio samples have a significantly lower speech naturalness compared to its neural network variants. Results published by Shen *et al.* (2017) show that conditioning WaveNet on mel-spectrograms predicted by Tacotron 2 achieves a mean opinion score (MOS) of 4.53 compared to a MOS of 4.58 for the ground truth audio samples.

In this work we explore the capability of Tacotron 2 and WaveRNN neural networks to synthesize speech in Romanian. To achieve this, we are going to use an open-source implementation of both Tacotron 2 and WaveRNN[2] developed as a project under Mozilla Common Voice, an initiative to develop the speech research domain. We use three Romanian speech datasets: (1) Romanian Speech Synthesis (RSS) corpus (Stan *et al.*, 2011), (2) Romanian Read Speech Corpus (RSC) (Georgescu *et al.*, 2020) and (3) SWARA Speech Corpus (Stan *et al.*, 2017).

The rest of this paper is composed as follows. The next section describes related work to our subject. Section 3 describes the training setup, the datasets and the details of the neural architectures used in the experiments. In section 4 we present the main results of our experiments with the Tacotron 2 and WaveRNN neural models for Romanian. Finally, in section 5 we present the key takeaways of the presented work.

2. Related work

Tacotron (Wang *et al.*, 2017) is a neural network architecture for speech synthesis directly from text. It uses <text, audio> pairs and it does not require phoneme-level alignment, which allows scaling to using large amounts of acoustic data with textual transcripts. Using a Griffin-Lim vocoder, it achieved a 3.82 mean opinion score on US English, outperforming parametric systems in terms of naturalness.

Tacotron 2 (Shen *et al.*, 2017) is the second iteration of the initially proposed TTS system, Tacotron. The neural network differs from the original Tacotron by replacing the "CBHG" (1-D convolution bank + highway network + bidirectional gated recurrent unit) stacks and gated recurrent unit (GRU) layers in the encoder and decoder with vanilla long short-term memory (LSTM) and convolutional layers. Furthermore, it replaces the Griffin-Lim algorithm with the neural network vocoder WaveNet. The

[2] https://github.com/mozilla/TTS (Last accessed 10.11.2020)

model achieved a mean opinion score (MOS) of 4.53 comparable to a MOS of 4.58 for professionally recorded speech, thus providing near-human performance.

WaveNet is a deep neural network for generating raw audio waveforms; it was the state-of-the-art of its times and achieved natural synthesized speech in English and Mandarin. The model is fully probabilistic and autoregressive, with the predictive distribution for each audio sample conditioned on all previous ones. The WaveNet model is trained on data of thousands of samples per second of audio, since its input consist of raw audio waveforms, which are signals with very high temporal resolution, at least 16,000 samples per second (Oord *et al.*, 2016).

WaveRNN (Wave Recurrent Neural Networks) is a single-layer recurrent neural network with a dual softmax layer that is equivalent in quality with WaveNet model. This recurrent neural network improves the speed of audio generation reaching up to 4 times faster than real time in comparison to WaveNet which generates audio at only 0.3 times real time. Furthermore, WaveRNN is proved to generate, in real-time, high fidelity audio on low-power mobile CPUs (Kalchbrenner *et al.*, 2018).

We are not aware of any Romanian TTS application based on any of the previous described neural networks. The current TTS system in Romanian that generates good quality is the HMM and DNN-based system with the STRAIGHT (Kawahara *et al.*, 2001) and WORLD (Morise *et al.*, 2016) vocoders proposed by Stan *et al.* (2011) and Stan *et al.* (2017), accessible as an online demo which allows generating voice with speakers from the SWARA dataset.

3. Proposed method

Our experimental setup[3] is based on open-source implementations of both Tacotron 2 and WaveRNN, maintained by *Mozilla Common Voice TTS* community on GitHub. We do not train any model on proprietary hardware, instead we use the resources available on *Google Colaboratory*, a hosted service with many types of GPUs available varying from Nvidia Tesla K80s, T4s, P4s and P100s. Google Colaboratory service regularly checks jobs activity and discourages long running jobs by disconnecting and deleting the virtual machine, hence we are limited to jobs running no longer than 8 hours. Because of this constraint we create training checkpoints at each 500th iteration and store them in a linked Google Drive.

3.1. Tacotron 2 implementation

For each training experiment we apply the same hyper-parameters. We use a reduction factor *r* and *batch size* that gradually decrease from $r = 3$ and batch size of 64 to $r = 2$ and batch size of 32, after 5,000 iterations. Reduction factor *r* is essentially a frame dropout that weakens local connections. The greater the reduction factor, the faster the model trains and the weaker the local dependency on autoregressive inputs. Our model setup accepts only data with a minimum length of 6 characters and a maximum length of 150 characters. The model minimizes the *L1 loss* and is optimized using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.0001. For audio processing

[3] Our GitHub repository: https://github.com/marzus555/ro_neural_tts (Last accessed 10.11.2020)

we use a sample rate of 22,050 Hz to resample the audio clips, a 1,024 ms window length, 275 ms hop length, pre-emphasis 0.98, silence trimming, and audio normalization within [-4,4].

Tacotron 2 is a complex neural model which is composed of different components with a well-defined purpose. The model components are: *Encoder, Decoder, Stop-net, Post-net*. The encoder component of Tacotron 2 plays the role of encoding the text input into a hidden feature representation. On our model, we use phoneme input representations as it has been proved to positively affect the training and generalization over different languages (Zhang *et al.*, 2019). The encoder output is consumed by the decoder by using an attention network. Finally, the predicted mel-spectrogram is represented by the addition of decoders' output and a residual computed by the post-net component. The stop-net component is responsible for predicting the end of synthesis and it is trained separately from the rest of the network to not allow its loss to influence the rest of the model. To complete the pipeline, the predicted mel-spectrogram can be further fed to a Griffin-Lim vocoder which transforms it into audio waveforms. Fig. 1 depicts Tacotron 2 based TTS system with each of its components highlighted.

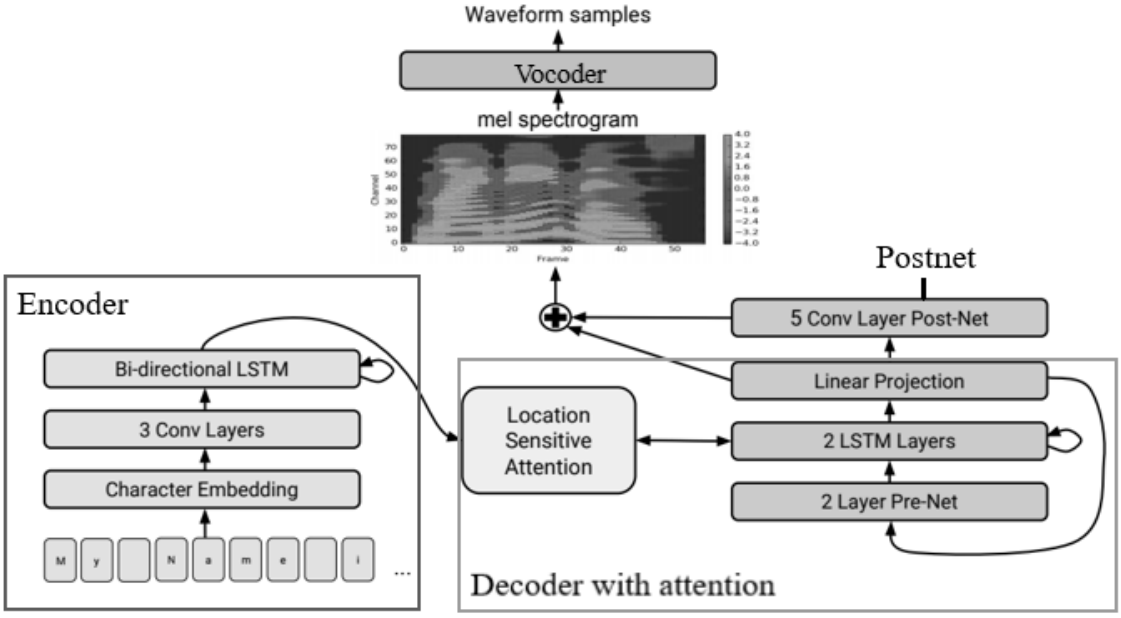


Figure 1: TTS system based on Tacotron 2 architecture (Shen *et al.*, 2017)

In order to start training the Tacotron 2 model on Romanian datasets we first needed to handle specificities of Romanian language. We have extended the model's character list to cover *ă, â, î, ș, ț* diacritics and their corresponding phonemes. We report results where the model is able to distinguish between words with and without diacritics. This proves that extending the list of characters with diacritics is crucial for a Romanian Tacotron 2 model. Working with our choice of datasets required trivial preparation to make them compatible with Tacotron 2. For our choice of Tacotron 2 implementation to start working with Romanian audio datasets, the only modifications required were to create the data mapping functionalities for each dataset to match the data feeding interface of Tacotron 2 and to extend the list of characters to include special characters of Romanian language.

3.2. WaveRNN implementation

To improve the quality of the synthesized speech, we switched from Griffin-Lim vocoder to WaveRNN neural network vocoder. The hyper-params used for WaveRNN model are the same as for Tacotron 2, except the audio normalization is done within $[0, 1]$ interval. The purpose of WaveRNN is to reconstruct high quality audio samples from mel-spectrograms. It achieves this by taking a mel-spectrogram as input and feeding it to an adapted gated recurrent unit (GRU) cell, where a single transformation produces all the required gates at once. The model is optimized using Adam by minimizing the negative log likelihood loss between the log-probabilities generated by the model and the *n*-bit quantized audio sample (Kalchbrenner *et al.*, 2018).

3.3. Datasets

RSS dataset is composed of 5,413 sentences from newspapers and short stories by Ion Creangă, uttered by 3 female speakers representing almost 6.5 hours of speech recorded at 96 kHz then down sampled to 48 kHz (Stan *et al.*, 2011). We allocated 90% of RSS dataset to the training set and the rest of 10% to the validation set.

SWARA Speech Corpus is one of the largest Romanian speech datasets summing up to almost 21 hours of speech coming from 17 speakers, both male and female (Stan *et al.*, 2017). For this dataset we used an 85% training set and 15% validation set split.

RSC is a speech corpus made up of 136k audio files collected from 164 Romanian native speakers, each speaker having between 130 and 11,000 utterances. Sentences in this dataset are represented by isolated words in Romanian language or text collected from literature and online news (Georgescu *et al.*, 2020). We split the data into 90% training set and 10% validation set.

4. Results

Determining whether the Tacotron 2 neural network is able to generate intelligible speech is not as straightforward as checking up if the model's loss is decreasing. Empirically, we have found out that a Tacotron 2 model is able to generate intelligible speech after a few thousands of iterations (the number of iterations differs from dataset to dataset) and an alignment score of over 0.5. For all our experiments and datasets, this empirical rule proved to be true in the ability of the model to generate speech. We encourage readers to listen to the audio samples generated by the models in Romanian and uploaded to our GitHub repository[4].

4.1. Experiments

Initially, we experimented with RSS and RSC datasets. On both datasets we experimented with two Tacotron 2 network variants, both with Griffin-Lim vocoder. The first variant is the baseline Tacotron 2, where all the audio samples are fed to the network without any additional information about the speaker, whereas the second variant is a multi-speaker model that feeds speaker identity information to the network.

[4] https://marzus555.github.io/ro_neural_tts/ (Last accessed 10.11.2020)

During our evaluation we noted that the baseline variant generates speech with arbitrary speaker identity from the speakers in the dataset, whereas the multi-speaker allows conditioning on speaker identity at inference time.

Table 1 shows information about the number of iterations and time to train for every experiment conducted on RSS and RSC datasets. The column *#iter.* represents the number of iterations for the experiment type, while the column *#hours* shows the corresponding hours required to reach the number of iterations from the previous column. Column *#int. mark* shows the iteration number from which the model started to generate intelligible speech. Models that generate intelligible speech but for which we did not record the intelligible speech mark are noted with "Not recorded".

Following our listening tests, we concluded that RSS trained model outperforms significantly the model trained on RSC, in terms of naturalness and speech quality. We compared generated audio samples of over 10 speakers from RSC dataset with the generated audio samples of 2 speakers from RSS dataset. All of the RSC audio samples have significantly lower naturalness than the RSS audio samples. Noteworthy is the ability of RSC multi-speaker trained model to generate speech with a male voice when conditioned on male speaker identity and similarly to generate speech with a female voice by conditioning it on a female speaker identity. Although the multi-speaker RSS trained model performs better in terms of speech quality, we note that it is able to generate speech only with 2 out of the 3 speaker identities available in the dataset.

Table 1: Experiment details on RSS and RSC datasets

Dataset / **Type of Experiment**	**RSS**			**RSC**		
	#iter.	#hours	#int. Mark	#iter.	#hours	#int. mark
Training from scratch	30,000	125	9,000	30,000	125	4,000
Multi-speaker training	20,000	83	Not recorded	20,000	83	Not recorded

In our next experiment we trained a multi-speaker Tacotron 2 on SWARA dataset. Table 2 shows information about the number of iterations, training hours and the iteration number when the model started to generate intelligible speech. This model not only generates the best quality from the three tested Romanian datasets, but it also allows speech synthesis from both male and female individuals. Similar to RSS trained model, we note that the network failed to model some of the speakers. Moreover, we observed that the quality of male generated samples is not on par with the female generated samples. It is important to note that the model generates the best quality only from setups that use the Griffin-Lim vocoder, but falls behind RSS dataset when the setup replaces Griffin-Lim vocoder with WaveRNN neural vocoder.

Table 2: Experiment details on SWARA dataset multi-speaker training

#iterations	#training hours	#intelligible speech mark
95,600	398	3,200

In Table 3, we show WaveRNN training information grouped by datasets and WaveRNN variants. We also distinguish between training on ground truth mel-spectrogram or Tacotron 2 generated mel-spectrograms. For both datasets and model variants, we use a *10-bits* quantization of the audio waveform. It was proved by authors of Tacotron 2 (Shen *et al.*, 2017) that their choice of neural vocoder (WaveNet) is performing better when trained on the Tacotron 2 predicted mel-spectrograms rather than training the neural vocoder on ground truth mel-spectrograms. This is due to predicted spectrograms being oversmoothed and less detailed than the ground truth. Unfortunately, we cannot prove if this applies to WaveRNN neural vocoder, since our training setups on the predicted mel-spectrograms failed for both SWARA and RSS datasets.

Table 3: WaveRNN experiments details on SWARA and RSS datasets

Dataset / **Type of Experiment**	**SWARA**			**RSS**		
	#iter.	#hours	#int. mark	#iter.	#hours	#int. mark
10-BITS Ground truth spectrograms	152,000	43	49,000	329,000	93	135,000
10-BITS Tacotron 2 spectrograms	118,000	33	No intelligible speech	190,000	54	No intelligible speech

We subjectively tested audio samples generated from models trained on both datasets and with both ground-truth and Tacotron 2 generated mel-spectrograms. Ground truth RSS trained model produces the most natural synthesized speech, while ground truth SWARA model produces good quality synthesized speech but with noise. We suspect the noise comes from the larger number of speakers. Moreover, the SWARA dataset includes audio samples coming from male speakers too, thus increasing the difficulty of the model to map mel-spectrograms to audio waveforms.

Furthermore, we found that the ground truth RSS trained model is able to synthesize speech from Tacotron 2 generated mel-spectrograms, which is of considerably higher quality than those synthesized with Griffin-Lim vocoder. This is an indication that the WaveRNN model does not have the same problem reported in WaveNet by the authors of Tacotron 2, although this is not conclusive since we do not have any results for the models trained on predicted spectrograms.

4.2. Mean opinion score evaluation

For speech synthesis there are no generally-accepted objective metrics to verify the quality of generated samples. Best way of evaluating the quality of synthesized speech is by listening to the audio samples. Mean opinion score (MOS) tests come in handy for audio sample subjective evaluation. We conducted a mean opinion score to subjectively evaluate the speech naturalness of our models. We randomly selected 11 sentences from the book "Iona" by Marin Sorescu that were not seen during training and we asked the subjects to evaluate only the naturalness of the generated voice with a rating from 1 (not natural) to 5 (natural). For each sentence, the subject was presented with 3 audio samples generated by different TTS setups. The first audio sample was generated by our Tacotron 2 model with the Griffin-Lim vocoder, the second audio sample was generated from RomanianTTS online demo[5] where the TTS system was trained on SWARA dataset, and the third audio sample was generated again by our Tacotron 2 model but with WaveRNN neural vocoder. Some of the audio samples proposed to evaluation presented white noise at the end of the clip. To allow the subjects to focus only on the naturalness, we have processed the audio samples only by trimming the white noise at the end. In terms of speech correctness, we report that none of the tested audio samples have any issues.

To provide an easy way for users to rate audio samples we have created a custom online application[6]. The samples were relatively evaluated to each other since the subjects had access to all the audio samples regardless of their TTS system. A total of 15 users have rated all of the audio samples proposed to evaluation. Table 4 shows the results of our MOS for the three TTS systems. Our MOS evaluation shows that Tacotron 2 with WaveRNN significantly outperforms both Tacotron 2 with Griffin Lim and RomanianTTS audio samples. In contrast, we observe that Tacotron 2 with Griffin Lim vocoder obtained the lowest rating out of all 3 systems.

Table 4: Mean opinion score evaluation results

System	MOS	Std. Deviation
Tacotron 2 + Griffin Lim	2.324	0.972
RomanianTTS with WORLD vocoder	3.732	1.018
Tacotron 2 + WaveRNN	4.197	0.833

5. Conclusions

We explored various Tacotron 2 and WaveRNN training approaches on Romanian datasets and presented the details of each approach. We proved the ease of creating a TTS pipeline for Romanian language with the help of Tacotron 2 and WaveRNN neural networks. The ability to generate good quality speech from the models trained on RSS, a dataset with just 6.5 hours of speech, is noteworthy. Multi-speaker Tacotron 2 models provided the best quality of synthesized speech and, in addition, allows the model to control the speaker identity at inference time. To achieve our results, we trained Tacotron 2 models between 83 hours and 125 hours for the RSS and RSC datasets. We

[5] http://romaniantts.com/ (Last accessed 10.11.2020)
[6] https://poli-text-audio.web.app (Last accessed 10.11.2020)

invested the most amount of training time, almost 400 hours, into the multi-speaker Tacotron 2 model trained on SWARA datasets, while for WaveRNN models we trained only between 33 hours and 93 hours.

We showed that our TTS system, composed of Tacotron 2 and WaveRNN neural network vocoder, outperforms the Griffin-Lim variant and the TTS system from RomanianTTS, achieving a 4.197 subjective 5-scale mean opinion score.

Future work will explore fine-tuning the multi-speaker models, both Tacotron 2 and WaveRNN, on a targeted speaker. This training approach aims to obtain a higher quality TTS system, especially for datasets with a greater number of speakers such as SWARA and RSC.

Finally, we plan on releasing an online TTS demo for Romanian that will use at its core Tacotron 2 and WaveRNN neural models trained on SWARA and RSS datasets, as both of these datasets are under a *Creative Commons* license.

References

Georgescu, A. L., Cucu, H., Buzo, A. and Burileanu, C. (2020). RSC: A Romanian Read Speech Corpus for Automatic Speech Recognition. In *Proceedings of The 12th Language Resources and Evaluation Conference*, 6606-6612.

Griffin, D. and Lim, J. (1984). Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 32(2),* 236-243.

Hunt, A.J. and Black, A.W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. In *1996 IEEE International Conference on Acoustics, Speech and Signal Processing Conference Proceedings,* IEEE, Vol. 1, 373-376.

Kalchbrenner, N., Elsen, E., Simonyan, K., Noury, S., Casagrande, N., Lockhart, E., Stimberg F., van der Oord, A., Dieleman, S. and Kavukcuoglu, K. (2018). Efficient neural audio synthesis. *arXiv preprint arXiv:1802.08435, https://arxiv.org/pdf/1802.08435.pdf, 1-10*.

Kawahara, H., Estill, J. and Fujimura, O. (2001). Aperiodicity extraction and control using mixed mode excitation and group delay manipulation for a high quality speech analysis, modification and synthesis system STRAIGHT. In *Second International Workshop on Models and Analysis of Vocal Emissions for Biomedical Applications,* 59-64.

Kingma, D.P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980,* https://arxiv.org/pdf/1412.6980.pdf, 1-15.

Morise, M., Yokomori, F. and Ozawa, K. (2016). WORLD: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE TRANSACTIONS on Information and Systems, 99(7),* 1877-1884.

Oord, A. V. D., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner N., Senior, A. and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499,* https://arxiv.org/pdf/1609.03499.pdf, 1-15.

Shen, J., Pang, R., Weiss, R.J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R. and Saurous, R.A. (2018). Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP, IEEE, 4779-4783.

Stan, A., Dinescu, F., Ţiple, C., Meza, Ș., Orza, B., Chirilă, M. and Giurgiu, M. (2017). The SWARA speech corpus: A large parallel Romanian read speech dataset. In *2017 International Conference on Speech Technology and Human-Computer Dialogue,* SpeD, 1-6, IEEE.

Stan, A., Yamagishi, J., King, S. and Aylett, M. (2011). The Romanian speech synthesis (RSS) corpus: Building a high quality HMM-based speech synthesis system using a high sampling rate. *Speech Communication*, 53(3), 442-450.

Taylor, P. (2009). *Text-to-Speech Synthesis*. Cambridge University Press.

Wang, Y., Skerry-Ryan, R.J., Stanton, D., Wu, Y., Weiss, R.J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S. and Le, Q. (2017). Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135,* https://arxiv.org/pdf/1703.10135.pdf, 1-10.

Zhang, Y., Weiss, R.J., Zen, H., Wu, Y., Chen, Z., Skerry-Ryan, R.J., Jia, Y., Rosenberg, A. and Ramabhadran, B. (2019). Learning to speak fluently in a foreign language: Multilingual speech synthesis and cross-language voice cloning. *arXiv preprint arXiv:1907.04448, https://arxiv.org/pdf/1907.04448.pdf, 1-5.*

ROMANIAN SPEECH RECOGNITION EXPERIMENTS FROM THE ROBIN PROJECT

ANDREI-MARIUS AVRAM, VASILE PĂIȘ AND DAN TUFIȘ

Research Institute for Artificial Intelligence, Romanian Academy

{andrei.avram,vasile,tufis}@racai.ro

Abstract

One of the fundamental functionalities for accepting a socially assistive robot is its communication capabilities with other agents in the environment. In the context of the ROBIN project, situational dialogue through voice interaction with a robot was investigated. This paper presents different speech recognition experiments with deep neural networks focusing on producing fast (under 100 ms latency from the network itself), while still reliable models. Even though one of the key desired characteristics is low latency, the final deep neural network model achieves state of the art results for recognizing Romanian language, obtaining a 9.91% word error rate when combined with a language model, thus improving over the previous results while offering at the same time an improved runtime performance. Additionally, we explore two modules for correcting the Automatic Speech Recognition output (hyphen and capitalization restoration and unknown words correction), targeting the ROBIN project's goals (dialogue in closed micro-worlds). We design a modular architecture based on application programming interfaces allowing an integration engine (either in the robot or external to it) to chain together the available modules as needed. Finally, we test the proposed design by integrating it in the RELATE platform and making the Automatic Speech Recognition service available to web users by either uploading a file or recording new speech.

Key words — automatic speech recognition, deep neural networks, natural language processing, Romanian.

1. Introduction

ROBIN[1] is a user-centred project designing software systems and services for the use of robots in an interconnected digital society. The project covers a diverse range of robots: assistive robots for the support of people with special needs, social robots for interaction with store customers, and software robots that can be installed on intelligent vehicles to achieve autonomous car driving.

One of the subprojects, called ROBIN-Dialog[2], aims to develop a series of scenarios for several micro-worlds, as well as to develop the technology of processing the Romanian language for situational dialogues in these micro-worlds. In this context, voice interaction with the dialog system of the robot becomes very important. The end-result involves linking together an automatic speech recognition (ASR) system, a dialog

[1] http://aimas.cs.pub.ro/robin/en/

[2] http://aimas.cs.pub.ro/robin/en/robin-dialog/

management system and a text to speech (TTS) system. This allows a user to interact with the robot using only spoken language.

Because the end system has several components (ASR, dialog component, TTS), each of them must exhibit low latency and execute in near real-time to improve the experience of the user. This paper focuses on ASR system experiments performed with the goal of achieving a latency as low as possible, while still obtaining state-of-the-art (SOTA) results. The module itself was developed and tested outside the dialog system, while providing application programming interfaces (API) that allows it to be integrated into other complex systems.

This paper is structured as follows: in Section 2 we present related work, including previous work within the ROBIN project, followed by the proposed modular system architecture in Section 3. Then, in Section 4 we describe the datasets used, followed in Sections 5 and 6 by a description of the ASR system and the implemented modules. Section 7 presents the evaluation results. Finally, the conclusions and possible future work directions are presented in Section 8.

2. Related work

ASR consists in translating human spoken utterances into a textual transcript, and it is a key component in voice assistants (Lopatovska *et al.*, 2019), in spoken language translation systems (Di Gangi *et al.*, 2019) or in generating automatic transcriptions for audio and videos (Noda *et al.*, 2014). Most of the ASR systems before the deep learning revolution used variations of Hidden Markov Models (HMM) (Garg and Sharma, 2016), and, although they achieved good Word Error Rates (WER), they became very slow for large vocabularies and could not be used for open domain real-time transcriptions. ASR systems can be classified into end-to-end and pipeline systems. The former can get both raw audio wave and handcrafted features as input, while pipeline systems have specific components (assembled into a processing pipeline) to extract speech features. Both system types can benefit from additional text correction modules.

Georgescu *et al*. (2019) considered the application of neural networks to Romanian ASR systems using the Kaldi3 toolkit. As described in the paper, the authors evaluated the model on two corpora: RSC-eval and SSC-eval, achieving WERs of 2.79% and 16.63%, respectively. Even though the two evaluation sets represent different types of speech (read speech and spontaneous speech), for our implementation we are interested in a general ASR system, working regardless of the speech type. Hence, for comparison purposes we consider the average WER of both evaluation corpora, thus leading to an average WER of 9.71%. Furthermore, the Kaldi toolkit uses a pipeline system, where each component is treated as an independent module, so, regarding latency, this approach has the disadvantage of each module adding its own latency to the overall process.

For the purposes of the ROBIN project, a similar approach based on the Kaldi toolkit was considered in Tufiș *et al*. (2019a). Preliminary results, reported in the paper, were rather modest, yielding a WER of approximately 25%, which is larger than the WER

[3] https://kaldi-asr.org

reported by Georgescu *et al*. (2019) on the SSC-eval corpus. Nevertheless, this can be attributed to the different audio corpora used for training the model, rather than to the technology itself.

One of the recent end-to-end speech recognition architectures is DeepSpeech2. According to Amodei *et al.* (2016), models trained with this toolkit were able to achieve a WER of less than 10% for both English and Mandarin on several evaluation datasets. To achieve this result, 11,940 hours of English speech data was fed through a deep neural network consisting of 11 layers, while for Mandarin the training dataset consisted of 9,400 hours. Additionally, in a previous research (Avram *et al.*, 2020), we also confirmed that this architecture is capable of obtaining a WER less than 10% for the Romanian language, while considering both read and spontaneous speech.

Comparing with the large volume of speech data used by Amodei *et al*. (2016), totalling around 10,000 hours for each of the investigated languages, data available within the ROBIN project for the Romanian language is far smaller, totalling around 230 hours of audio aligned with text. Nevertheless, we considered experimenting with only this amount of data, trying to construct an end-to-end ASR system for Romanian.

3. System architecture

Given the goal of integrating the resulting ASR system into the larger ROBIN-Dialog context, the implementation needs to encapsulate the functionality into a dedicated module and expose it via easy-to-use APIs. Furthermore, taking into account that envisaged human-robot dialogues are well defined, based on a closed-world scenario, controlled by a Dialog Manager (DM) (Ion *et al.*, 2020), the ASR system can be complemented by different correction models, further improving the recognized speech. These correction models are also exposed as APIs and invoked as needed.

For API invocation we considered using the HyperText Transfer Protocol (HTTP) GET and POST requests. Furthermore, the implementation adheres to a stateless, client-server, Representational State Transfer (REST) design. Finally, the result of an API call will be provided in JSON format. Thus, invoking a method involves sending a standard HTTP request with the required parameters sent according to the specific method (GET or POST) and parsing the output JSON to extract the results. Since high-performance JSON parsing libraries exist for most of the programming languages, this encoding will not add significant latency to the overall process.

The following modules with their corresponding methods and parameters were envisaged:

a) ASR: "/transcribe" method uses a single parameter: "file". The method receives a WAV file in the parameter via the POST HTTP request, and then it invokes the ASR model to transcribe the received file. Finally, it returns a JSON with two fields: "status", representing the success or error of the call, and the corresponding transcription in the "transcription" field.

b) Hyphen restoration and capitalisation: "/correct" method receives a "text" parameter via either HTTP GET or POST. The received text is supposed to be the result of the ASR module "/transcribe" method described above. The method checks for missing

hyphens and tries to restore them. The output JSON contains a "status" field representing the success or error of the call, a "text" field with the corrected text and an optional "comments" field used for analysing the correction process (not needed for integrations).

c) Additional language model-based correction for unknown words: "/correct" method behaving similarly to the previous method, receiving a "text" parameter and returning an output JSON with "status", "text" and optional "comments".

We opted for an architecture with low cohesion, so we isolated each module and created a chain of API calls that are invoked by the integration system, as depicted in Figure 1. This design was considered because it also allowed the integration system to select the calls and their order depending on the context. For example, it can be considered that very short transcriptions, consisting of only one or two words may not need hyphen and capitalization correction, while they may still employ other corrections. Additionally, the design allows evaluation and timing of each individual module.

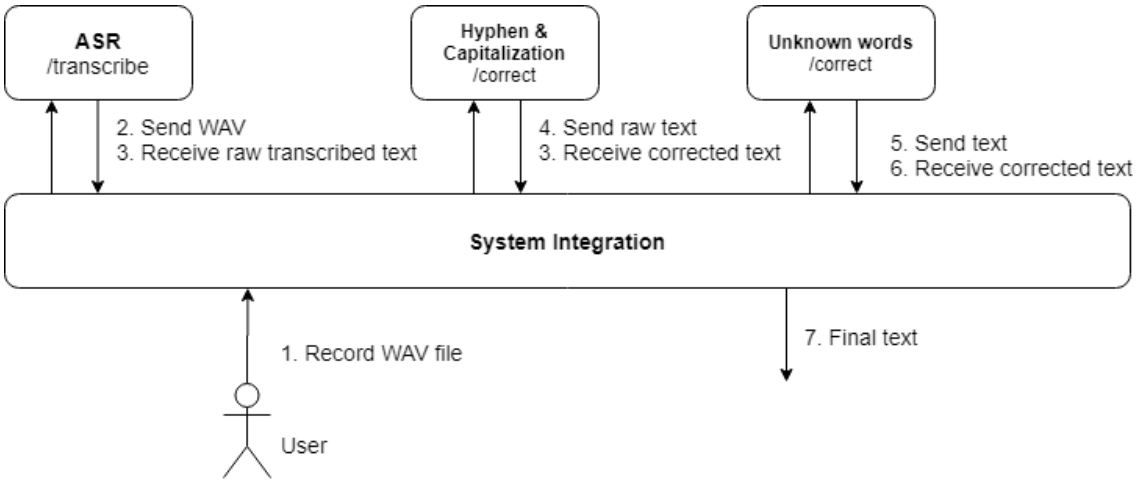


Figure 1: System integration architecture

4. Datasets

Two kinds of corpora were needed for the purposes of this work: first, a multimodal corpus containing high quality alignment of speech to text that was used to train the actual ASR system; second, large text corpora that were used to train the different language models for implementing additional corrections.

The main audio resource used was the speech component of the representative corpus of the contemporary Romanian language (CoRoLa) (Tufiș *et al*., 2019b). CoRoLa has been jointly developed, as a priority project of the Romanian Academy, by two institutions: "Mihai Drăgănescu" Research Institute for Artificial Intelligence (from Bucharest) and the Institute of Computer Science (from Iași). The oral texts in CoRoLa are mainly professional recordings from various sources (radio stations, recording studios). They are accompanied by the written counterpart: the transcription, either from their provider or made by the project partners. Therefore, different principles applied in their transcription, potentially making it more difficult to use for the purposes of the ASR

system. Another part of the oral corpus is represented by read texts: read news in radio stations, texts read by professional speakers recorded in studios, and extracts from Romanian Wikipedia read by non-professionals, by volunteers, recorded in non-professional environments. In their case, the written component is provided by the sources, or was collected by the project partners (Mititelu *et al*., 2018).

The speech component of the CoRoLa corpus can be interrogated by means of the Oral Corpus Query Platform (OCQP)[4]. This allows searching for words and listen to their spoken variant, based on the alignment between text and speech (Tufiș *et al*., 2019b).

In the context of the RETEROM project[5], the CoBiLiRo platform (Cristea *et al*., 2020) was built to allow gathering of additional bimodal corpora with one of the final goals being to enrich the CoRoLa corpus. Thus, additional corpora with speech and text alignments were employed. This includes: Romanian Digits (RoDigits) (Georgescu *et al.*, 2018), Romanian Speech Synthesis (RSS) (Stan *et al*., 2011), Romanian Read Speech Corpus (RSC) (Georgescu *et al*., 2020).

Additionally, the Common Voice corpus (Ardila *et al.*, 2019) was considered, as it is a massively multilingual dataset of transcribed speech that, as of October 2020, contains over 7,300 hours of validated audio in 54 languages from over 50,000 speakers. The Romanian version is one of the recently added languages and its corresponding corpus contains 7 hours of transcribed audio recorded by 79 speakers, from which only 5 hours are validated. The corpus sentences were collected from Wikipedia using a sentence collector, and each sentence must be approved by two out of three reviewers before reaching the final version of the corpus, thus allowing for an acceptable audio quality aligned with the short text representing the sentence.

The main text resource used for language models was represented by the CoRoLa corpus. It is a large, growing collection of Romanian texts, currently containing 941,204,169 tokens. Currently, all texts are more recent than 1945, therefore CoRoLa is a contemporary corpus. Various annotation levels were employed and the corpus can be queried through various interfaces (Cristea *et al.*, 2019), including KorAP (Banski *et al*., 2012).

In addition to the texts from CoRoLa, we considered the OSCAR corpus (Suárez *et al.*, 2019). It is a huge open-source multilingual corpus that was obtained by filtering the Common Crawl[6] and by grouping the resulting text by language. The Romanian version contains approximately 11 GB of deduplicated shuffled sentences. Even though CoRoLa is a representative corpus of the Romanian language, we considered that adding more text to the training of a language model could benefit in terms of accuracy.

[4] http://corolaws.racai.ro/corola_sound_search/index.php
[5] http://www.racai.ro/p/reterom/
[6] https://commoncrawl.org/

5. The ASR system

Following the DeepSpeech2 architecture, we first computed the Mel-frequency cepstral coefficients (MFCC) (Logan, 2000) on fixed-size audio windows of 20 ms. Then, the resulted spectrogram was fed into a deep neural network model. However, since the model presented by (Amodei *et al.*, 2016) made use of a large number of speech hours we considered a simpler model, consisting of only 8 neural layers: 2 convolutional 2D layers, 4 layers with bidirectional long short-term memory (BiLSTM) (Hochreiter and Schmidhuber, 1997), 1 lookahead convolution and a final fully connected (FC) layer. Batch normalization (BN) (Ioffe and Szegedy, 2015) was then used after each layer except the last one for faster model convergence and more stability during training. Also, we used the Hard Hyperbolic Function (HardTanh) activation function before each (BN), a cheaper and more computationally efficient version of tanh, as proposed in (Xiang *et al*., 2017).

The output of the network is a vector of size 33 that is computed using the FC layer and the softmax activation and it represents a distribution of probabilities over the 31 Romanian characters, together with the space character and the blank index. The "blank" index is a special character used for cases when a certain character is repeated. For example, "acceptat" is encoded as "ac_ceptat" to mark not to collapse the character "c" while decoding. Furthermore, to account for situations where the utterance of a character may take more than the size of a window, the Connectionist Temporal Classification (CTC) (Graves *et al*., 2006) loss was used to train the network. Additionally, the Adaptive Moment Estimation (Adam) optimizer (Kingma and Ba, 2015) was used, initially with a high learning rate to accelerate the training, and then applying a learning rate decay of 5% after each epoch to avoid oscillations in the later stages of training (You *et al*., 2019).

Following the methodology in Amodei *et al*. (2016), we incorporated a 5-gram language model for transcription correction in the ASR system, that was trained using the KenLM toolkit[7]. The toolkit allows the configuration of two hyperparameters: alpha (α), which controls the contribution of the model to predicting a word, and beta (β) – the probability of inserting a new word into the sequence. Thus, during inference we search for the most probable transcription using (1):

$$Q(y) = \log(p_{ctc}(y|x)) + \alpha \log(p_{lm}(y)) + \beta word\_count(y) \qquad (1)$$

where x is the audio input, y is the output of the neural model, p_{ctc} is the probability given by the CTC decoder, p_{lm} is the probability given by the language model, $word\_count$ is the number of words in the predicted transcription, and α and β are the hyperparameters of the language model.

For the purposes of building this language model, the available text resources were first pruned to remove potentially wrong data, especially with regards to the OSCAR corpus. This involved applying removal rules such as: removing very short (less than 20 characters) or very long lines, lines with words containing no diacritics, lines with URLs. Additionally, known abbreviations and measurement units were replaced with

[7] https://github.com/kpu/kenlm

their complete textual representation. This resulted in about 10 GB of "cleaned" raw text from the combination of CoRoLa and OSCAR.

The resulting system was transformed into a REST API server using the Flask framework[8] and the Waitress web server[9], according to the API specification described in Section 3. The service makes use of a configuration file, allowing tuning of the system parameters such as the beam width, whether to use or not the language model, or whether to use a GPU or only run on the CPU. The expected "file" parameter must represent a WAV file with the following characteristics: mono, 16-bit, 16 KHz.

6. Additional text corrections

As described in Section 3, several text correction mechanisms for the ASR output were envisaged. These aim to improve the results of the speech recognition system. They are based on n-gram models, trained on the CoRoLa corpus.

Since our neural network model generates characters (and not complete words), this may include situations were a hyphen is normally used, but it is not present in the output. In some cases, such as "ți-am" vs. "țiam" the correction is easy to perform, since "țiam" is not a valid Romanian word. However, in cases such as "s-a" vs. "sa" the correction is more difficult since both words are valid in Romanian. Therefore, a more complex approach, based on context, is needed. For this purpose, we first employed a bigram model considering the frequency of using the current word (W_k) with and without hyphen together with the next word (W_{k+1}). If for some reason the bigram (W_k, W_{k+1}) is not available (possibly due to the following word being recognized incorrectly) we fall back to a unigram model, replacing the current word (W_k) with the most frequent form.

Basic capitalization restoration is performed using name lists. We considered reduced name lists containing mostly people names and locations (countries, large cities) to reduce the risk of capitalizing a word simply because it looks like a named entity. As opposed to named entity recognition, in this case we do not need to actually identify in text the corresponding entity type since, regardless of the type, person names and location names will be written with first letter capitalized.

Given that other recognition errors, apart from hyphens, may happen, we further considered an additional correction model making use of the Levenshtein distance to correct unknown words. For this purpose, we consider unknown the words not appearing in the vocabulary generated from the CoRoLa corpus, considering a minimum number of occurrences of 10. When an unknown word Wk is encountered, the system will look for a bigram (W'_k, W_{k+1}) or (W_{k-1}, W'_k) having the Levenshtein distance between W_k and W'_k less than a certain threshold. If no bigram is identified, we fall back to a unigram model, checking the Levenshtein distance with other words in the vocabulary, having similar size (less than the considered threshold).

[8] https://flask.palletsprojects.com/en/1.1.x/
[9] https://docs.pylonsproject.org/projects/waitress/en/stable/

The text corrections were implemented in two modules: one for hyphen restauration and capitalization and one for the unknown words. Both were implemented in Java and exposed as REST APIs according to the description given in Section 3.

7. Results

For system evaluation we used a server with a Xeon 4210 CPU running at 2.2 GHz and a single Quadro RTX 5000 GPU. As presented in the Introduction, we were interested in obtaining a model with low runtime latency. The ASR system's average recognition time on audio samples of less than 25 seconds was 600 ms, when running on the CPU, and 70 ms, when running on the GPU. With regard to space requirements, the DeepSpeech2 model file size is 160 MB and the uncompressed KenLM file size is 10.6 GB, both models occupying around 5.9 GB in RAM.

From the overall speech corpus, we extracted 5,000 samples that were used as the test set and another 5,000 samples that were used as the development set. All the audio files in both sets were less than 25 seconds in length, similar to the maximum expected ROBIN interactions.

For the purposes of the first language model employed in combination with the raw ASR, we used a grid search on the development dataset to find the best values, for α in the [0, 1.5] interval and for β in the [0, 3] interval with discrete steps of 0.1. The results of the grid search are depicted as a surface plot in Figure 2. As it can be observed, α has the highest influence on the overall WER, while β acts just as a regularizer for the predicted transcription, having a lower influence. The optimum for the hyperparameters is represented by the values of 0.3 for α and 1.5 for β. Using the combined ASR and language model (with the optimal parameters) the overall result was 9.91% WER on the test set, improving the WER of the raw ASR system with 5.66%.

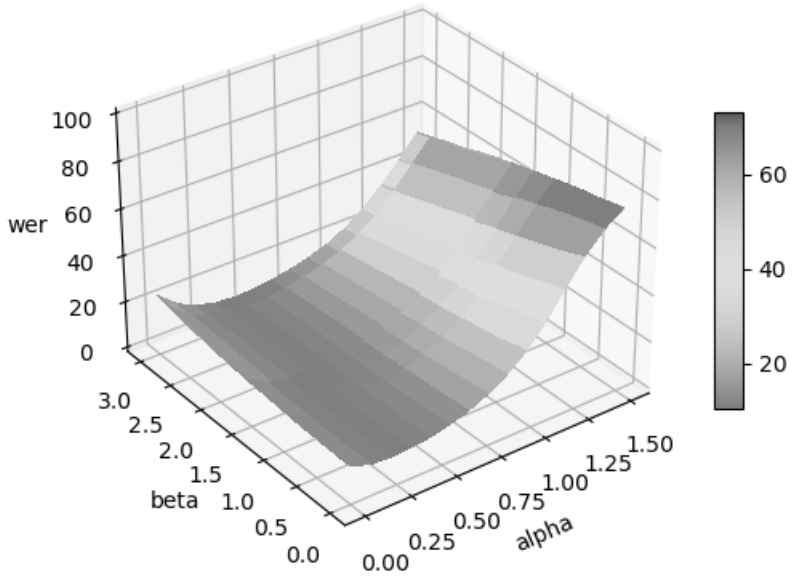


Figure 2. Surface plot of the WER with respect to the KenLM hyperparameters.

The other correction components were created mainly for the purposes of the ROBIN project, considering the envisaged interaction scenarios (short questions associated with closed worlds). Therefore, they have not been evaluated on the general ASR test set and it is envisaged to be later evaluated on a dedicated ROBIN set. Nevertheless, the architecture presented in Section 3 is valid and any improvements can be realized at module level. Furthermore, in order to test the integration capabilities, the modules were integrated in the RELATE platform (Păiș *et al.*, 2019), allowing users to upload a recorded wav file or make a recording directly in the platform and run it through the ASR system. The predicted transcription can then be analysed using the available annotation mechanisms within the RELATE platform. A picture of the Robin ASR in the RELATE platform is depicted in Fig. 3.

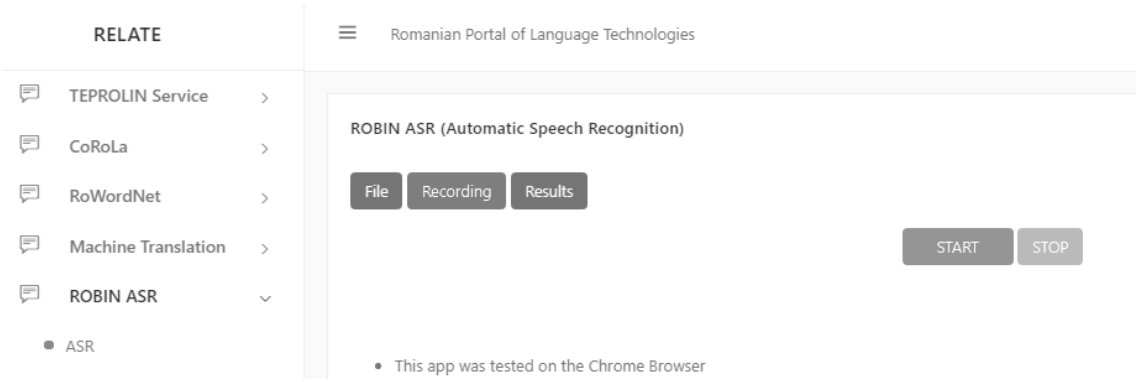


Figure 3. Picture of Robin ASR system in the RELATE platform.

8. Conclusions and Future Work

This paper presents the results of experimenting with an automatic speech recognition system for Romanian language, in the context of the ROBIN project. Our main goal was to obtain a low latency system, usable in near real-time applications such as voice interaction with a robot. However, in addition to the low latency (70 ms average recognition time) we managed to obtain a system with very good performance, 9.91% WER on our balanced test split.

The system was exposed as a REST API service, complemented by additional text correction modules, also available as individual services. The proposed integration scheme into more complex systems was validated by a first integration in the RELATE platform. The code is available on RACAI's GitHub account[10] together with the open-sourced release of the dialog manager[11] (Ion *et al.*, 2020).

As future work, we consider integrating the Romanian BERT (Dumitrescu *et al.*, 2020) in the postprocessing pipeline and use the model as a replacement for the n-gram language model to correct the transcription inferred by the ASR module, as presented in (Hrinchuk *et al.*, 2020). However, the integration of BERT might add considerable

[10] https://github.com/racai-ai/RobinASR
[11] https://github.com/racai-ai/ROBINDialog

latency overhead due to its high computational cost, so a pruning technique should be applied beforehand.

Acknowledgements

This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS – UEFISCDI, project number PN-III 72PCCDI/2018, ROBIN – "Robots and Society: Cognitive Systems for Personal Robots and Autonomous Vehicles".

References

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G. and Chen, J. (2016). Deep speech 2: End-to-end speech recognition in English and Mandarin. In *International conference on machine learning*, New York City, USA, 173-182.

Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F.M. and Weber, G. (2019). Common voice: A massively multilingual speech corpus. *arXiv:1912.06670*.

Avram, A.-M., Păiș, V., Tufiș, D. (2020). Towards a Romanian end-to-end automatic speech recognition based on DeepSpeech2. In *Proceedings of the Romanian Academy*, Series A, in-print.

Bański, P., Fischer, P., Frick, E., Ketzan, E., Kupietz, M., Schnober, C., Schonefeld, O., Witt, A. (2012). The New IDS Corpus Analysis Platform: Challenges and Prospects. In *Proceedings of the 8th International Conference on Language Resources and Evaluation* (LREC 2012), 2905-2911.

Barbu Mititelu, V., Tufiş, D. and Irimia, E. (2018). The reference corpus of the contemporary Romanian language (CoRoLa). In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, Miyazaki, Japan, 1178-1185.

Cristea, D., Diewald, N., Haja, G., Mărănduc, C., Mititelu, V.B., Onofrei M. (2019). How to find a shining needle in the haystack. Querying CoRoLa: solutions and perspectives. *Revue Roumaine de linguistique*, LXIV (3), 279-292

Cristea, D., Pistol, I., Boghiu, Ș., Bibiri, A.D., Gîfu, D., Scutelnicu, A., Onofrei, M., Trandabăț, D., Bugeag, G. (2020). CoBiLiRo: A Research Platform for Bimodal Corpora. In *Proceedings of the 1st International Workshop on Language Technology Platforms (IWLTP 2020)*, Language Resources and Evaluation Conference (LREC 2020), Marseille, France, 22-27.

Di Gangi, M.A., Negri, M. and Turchi, M. (2019). Adapting Transformer to end-to-end spoken language translation. In *Proceedings INTERSPEECH*, Graz, Austria, 1133-1137.

Dumitrescu, S.D., Avram, A.M. and Pyysalo, S. (2020). The birth of Romanian BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, Association for Computational Linguistics, 4324-4328.

Garg, A. and Sharma, P. (2016). Survey on acoustic modeling and feature extraction for speech recognition. In *The 3rd International Conference on Computing for Sustainable Global Development*, New Delhi, India, 2291-2295.

Georgescu, A.L., Caranica, A., Cucu, H. and Burileanu, C. (2018). RODIGITS-a Romanian connected-digits speech corpus for automatic speech and speaker recognition. *University Politehnica of Bucharest Scientific Bulletin*, Bucharest, Romania, 45-62.

Georgescu, A., Cucu, H., Burileanu, C. (2019). Kaldi-based DNN Architectures for Speech Recognition in Romanian. In *Proceedings of the 2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, Timisoara, Romania, 1-6.

Georgescu, A.L., Cucu, H., Buzo, A. and Burileanu, C. (2020). RSC: A Romanian Read Speech Corpus for Automatic Speech Recognition. In *Proceedings of the 12th Language Resources and Evaluation Conference*, Marseille, France, 6606-6612.

Graves, A., Fernández, S., Gomez, F. and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, Pittsburgh, United States, 369-376.

Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 1735-1780.

Hrinchuk, O., Popova, M. and Ginsburg, B. (2020). Correction of Automatic Speech Recognition with Transformer Sequence-To-Sequence Model. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7074-7078.

Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International Conference on Machine Learning*, 448-456.

Ion, R., Badea, V.G., Cioroiu, G., Barbu Mititelu, V., Irimia, E., Mitrofan, M., Tufiș, D. (2020). A Dialog Manager for Micro-Worlds. *Studies in informatics and control*, 29:4, 411-420.

Kingma, D.P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations ICLR*.

Logan, B. (2000). Mel frequency cepstral coefficients for music modeling. In *Proceedings of the International Symposium on Music Information Retrieval ISMIR*, 1-11.

Lopatovska, I., Rink, K., Knight, I., Raines, K., Cosenza, K., Williams, H., Sorsche, P., Hirsch, D., Li, Q. and Martinez, A. (2019). Talk to me: Exploring user interactions with the Amazon Alexa. *Journal of Librarianship and Information Science*, United Kingdom, 984-997.

Noda, K., Yamaguchi, Y., Nakadai, K., Okuno, H.G. and Ogata, T. (2014). Audio-visual speech recognition using deep learning. *Applied Intelligence*, 42, 722-737.

Păiș, V., Tufiș, D. and Ion, R. (2019). Integration of Romanian NLP tools into the RELATE platform. In *Proceedings of the International Conference on Linguistic Resources and Tools for Processing Romanian Language – CONSILR 2019*, 181-192.

Stan, A., Yamagishi, J., King, S. and Aylett, M. (2011). The Romanian speech synthesis (RSS) corpus: Building a high quality HMM-based speech synthesis system using a high sampling rate. *Speech Communication*, 442-450.

Suárez, P.J.O., Sagot, B. and Romary, L. (2019). Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures. In *7th Workshop on the Challenges in the Management of Large Corpora*, (CMLC-7), Jul 2019, Cardiff, United Kingdom. 10.14618/IDS-PUB9021. hal-02148693.

Tufiș, D., Barbu Mititelu, V., Irimia, E., Mitrofan, M., Ion, R. and Cioroiu, G.(2019a). Making Pepper Understand and Respond in Romanian**.** In *Proceedings of the 22nd International Conference on Control Systems and Computer Science*, 682-688.

Tufiș, D., Barbu Mititelu, V., Irimia, E., Păiș, V., Ion, R., Diewald, N., Mitrofan, M., and Onofrei, M. (2019b). Little strokes fell great oaks. Creating CoRoLa, the reference corpus of contemporary Romanian. *Revue Roumaine de linguistique*, LXIV (3), 227-240.

Xiang, X., Qian, Y. and Yu, K., (2017). Binary Deep Neural Networks for Speech Recognition. In *Proceedings INTERSPEECH*, 533-537.

You, K., Long, M., Wang, J. and Jordan, M.I. (2019). How Does Learning Rate Decay Help Modern Neural Networks?. *arXiv:1908.01878*.

IMPROVED TEXT NORMALIZATION AND LANGUAGE MODELS FOR SPEED'S AUTOMATIC SPEECH RECOGNITION SYSTEM

CRISTIAN MANOLACHE[1], ALEXANDRU-LUCIAN GEORGESCU[1], HORIA CUCU[1], VERGINICA BARBU MITITELU[2] AND CORNELIU BURILEANU[1]

1 Speech and Dialogue Research Laboratory, University Politehnica of Bucharest, {cristian.manolache,lucian.georgescu,horia.cucu,corneliu.burileanu}@upb.ro
2 Romanian Academy Research Institute for Artificial Intelligence, vergi@racai.ro

Abstract

Automatic speech recognition (ASR) systems that use word-based language models require periodical updates to include new named entities (*e.g.* coronavirus, COVID-19) or collocations. Moreover, in particular for the Romanian language, the new hyphenated words pose additional problems. In this context, our study presents SpeeD's efforts in collecting new text corpora and using them for language modelling in the context of ASR. We also present the improvements made in the text normalization module to address the problems posed by hyphenated words. We evaluate the resulting language models both in terms of their ability to predict future words (perplexity and out-of-vocabulary rate) and in terms of their usefulness in ASR (word error rate). We report ASR relative improvements of around 10% for spontaneous speech, with small degradations for read speech.

Keywords — Automatic Speech Recognition, Language Model, Natural Language Processing, Romanian, text corpus.

1. Introduction

In recent years, there has been an increase in the development of automatic speech recognition (ASR) systems (Georgescu *et al.*, 2019), as a result of the increasing penetration of various types of robots in human activities (Tufiș *et al*., 2019), of devices and gadgets into everyday life, etc.

When targeting a specific domain (*e.g.* medicine, technology, news), the ASR system can be adapted to better suit that particular domain, thus increasing its accuracy (Cucu *et al.*, 2014). More specifically, the language model (LM) inside the ASR system can be adapted to the target domain or even trained from scratch on domain-specific text. Depending on the size of the specific text corpus available, the word error rate (WER) of a generic ASR system can be lowered, through LM adaptation, by as much as 30% (Cucu *et al.*, 2014).

However, even if not targeting a niche domain, the LM inside a generic ASR system requires periodic updates in order to be able to model named entities, expressions or words that have appeared recently and/or were not modelled initially (*e.g.* "SARS-CoV-2", "COVID-19", "Dominic Fritz", "Emmanuel Macron", etc.). Moreover, it is necessary to periodically update the statistics for named entities or expressions that appear less (*e.g.* "anexarea Peninsulei Crimeea" (En. "Crimean Peninsula annexation"),

"criza financiară globală" (En. "global financial crisis")) or more often in daily speech (*e.g.* "Organizaţia Mondială a Sănătății" (En. "World Health Organization")).

The goal of this paper is to present the efforts made by Speech and Dialogue (SpeeD) Laboratory to update the language models powering its ASR systems. The contributions of this paper are as follows. First, we describe the process of collecting a new text corpus comprising recent news and present statistics about it. Second, we identify hyphen-related speech transcription issues, propose a solution and present the updates made to the text preprocessor in order to address the issues. Third, we develop new language models, tune their hyperparameters and evaluate and compare them in terms of their ability to predict future words (perplexity and out-of-vocabulary rate) and in terms of their usefulness in ASR (word error rate). Finally, we provide state-of-the-art results on three Romanian speech recognition evaluation datasets.

The rest of the paper is organized as follows: Section 2 covers the acquisition process of a new text corpus, which has been used to develop new language models. Section 3 describes the improvements brought to the Natural Language Processing (NLP) tool used for the normalization of the acquired text. In Section 4 we present the experimental setup with details regarding the datasets used for the evaluation of the new LM and ASR systems, as well as a baseline LM and ASR system. In Section 5, the new LMs and ASR systems presented in Section 4 are compared to the baseline LM and ASR system. Finally, in section 6 we state our conclusions.

2. New text corpus

The online environment is a real source of content, both textual and multimedia: press articles, interviews or reports are posted daily, all containing words, expressions and phrases from everyday speech. Languages are in a continuous development; new words appear constantly, most often in the realm of proper names: both persons and various entities. Because of this, the collection of new sets of texts is important in order to keep an ASR system up-to-date. The LM is one of the fundamental components of such a system, having the role of combining words into sentences, based on the probability of consecutive word sequences encountered in the training set. An LM whose vocabulary is updated will allow the system to transcribe speech in which those new words are spoken.

This section presents the steps taken to collect and preprocess a new text dataset, further called **news2020** (~255 million words).

2.1. Method of acquisition

A new text corpus, which consists of written articles, has been collected from several Romanian news websites and publications by using a Java application. This application continuously ran between 26.06.2018 and 22.05.2020, periodically checking the RSS feeds of the news websites and downloading new articles found. Each news source was extracted in a separate thread, thus increasing the speed of the extraction process which consists of scraping the HTML content, downloading the page and looking for certain tags containing text of interest.

2.2. Corpus processing

In order to be used for creating new language models, the acquired corpus has undergone a series of processing operations. The text files obtained from the news websites are not ready to use for several reasons: some articles might be downloaded several times, the text could be incorrectly extracted from the html or it could have issues in itself: not written with correct diacritics or even not in Romanian. The initial processing of the corpus consisted in filtering operations, *i.e.* removing files which:

- have a size smaller than 500 bytes;
- are duplicates. This is done by comparing the download link associated with each file with the download links of the other files;
- have a percentage of diacritics (diacritics rate) smaller than a threshold of 5%.

After these processes have been completed, the now filtered files are all concatenated into one file corresponding to the source group. Next, the concatenated files were further processed by our NLP tool, which underwent some improvements (see Section 3). These processes have the purpose to normalize the text, by giving it a form suitable for the creation of new LMs. We are interested in obtaining a final text that is correct, composed exclusively of words, formed in their turn by characters that correspond to the phonemes of the language, while the numbers and special characters must be converted into their textual form.

One of the first such processes to be applied is the diacritics restoration. The Romanian language has words that contain some special characters with diacritics, and even if the texts are taken from the written press and should be written in standard and correct way, there are sometimes omissions of diacritics. Therefore, the texts have been passed through our automatic system for diacritics restoration (Iordache *et al.*, 2019).

Another procedure that was applied involves replacing URLs and emails with their spoken form. Usually, they contain strings such as "www" or symbols like ".", which is pronounced "punct" in Romanian, or "@" which is pronounced "a rond" in Romanian. Over time, our group has developed a software application (Cucu, 2011) that converts numbers and digits into their textual form in Romanian. It was also used in this case, applying this transformation to the newly acquired text. Abbreviations are another issue in text processing. Even if they are used in written form, they correspond to the pronunciation of the entire word in the spoken version. Therefore, we have expanded the abbreviations to the complete form of the words. As we specified earlier, special characters were replaced by their spoken form. For example, "$" becomes the word "dolar" (En. "dollar"), while "%" becomes the words "la sută" (En. "per cent").

The last procedure applied was to transform all characters to lowercase. ASR systems usually output mere text (no digits, punctuation marks or special characters), and their vocabularies contain a single version of the words, because it is inefficient to keep them in both uppercase and lowercase format.

3. Natural language processor updates

3.1. ASR errors in the context of hyphenated words

We have recently analysed the errors made by the ASR system, with special focus on hyphenated words, as they were involved in many of the found errors. This category is, however, heterogeneous. Several types can be distinguished and treated accordingly:

i. There are many cases in which the ASR hypothesis lacks the hyphen in the case of phonologically independent tokens: words with pronominal clitics (*e.g.* "ratat-o" (En. "missed it"), "adresându-i" (En. "addressing him/her"), "dați-ne" (En. "give us")), abbreviations with definiteness markers (the Romanian enclitic definite article) (*e.g.* "IPGR-ul", "TVA-ului"), suffixed abbreviations (*e.g.* "PDL-ist"), suffixed abbreviations with plural marker (*e.g.* "PSD-iști"), suffixed abbreviations with definiteness markers (*e.g.* "PDL-istul", "UNPR-istului"), borrowings with Romanian inflection markers (*e.g.* "site-uri" (En. "sites")). Writing them without a hyphen violates the language orthographical norms.

ii. Another case is represented by compound words such as "Turnu-Severin", "lateralo-ventral", "sud-americane", "prim-ministru", "social-democrați", "greco-catolici", "alba-neagra", "azi-noapte", "nord-estul", "reality-show", "all-inclusive", or approximations like "două-trei". Leaving the hyphen aside in such cases is also considered an ASR error.

iii. Similarly, there are prefixed words, such as "pro-guvernamentale", "co-finanțat", "sub-secretarul", "vice-președinte", "arhi-cunoscut", "super-eroi", which are acceptable both with or without a hyphen, in the latter case being written as a single word. An ASR output with or without hyphen should be considered correct, regardless of the word form occurring in the reference text.

iv. Finally, there are cases which are correct both with and without a hyphen. The presence of the hyphen indicates a faster speech rhythm, in which the two words were uttered without any pause in between: *e.g.* "și-am" versus "și am", "ce-a" versus "ce a", "nu-l" versus "nu îl", "ce-au" versus "ce au". An ASR output with or without hyphen should be considered correct, regardless of the word form occurring in the reference text.

For errors of types (i) and (ii), we describe our solution in the next subsection. For errors of type (iii) and (iv) the ASR evaluation application should be updated to accept both forms (*i.e.* with and without hyphen), but this is outside the scope of this paper.

3.2. Solution to the problem of missing hyphens in ASR output

ASR errors of type (i) and (ii) described in the previous section had their root cause in the natural language processor used to preprocess the text that would further be used for language modelling, as shown in the left part of Fig. 1.

The old natural language processor searched for hyphenated words in a dedicated lexicon; if the word was found in the lexicon, it was left unchanged (*e.g.* "să-mi", "să-i", "m-a"), otherwise the hyphen was replaced with a space (*e.g.* "lovit o", "mi aduci", "lăsându se", "FMI ul", "TVA ul"). The hyphenated words lexicon did not contain all

the possible hyphenated words because this would have meant an exponential growth of the size of the vocabulary for the LM inside the ASR system. For example, for each acronym such as "SMURD", several other forms would have been inserted in the vocabulary: "SMURD-ul", "SMURD-ului", etc. An even more problematic example is that many verb forms (*e.g.* gerunds, imperatives) would require many additional hyphenated words in the lexicon: for "lăsând" we would also need "lăsându-mă", "lăsându-te", "lăsându-l", "lăsând-o", "lăsându-ne", "lăsându-vă", "lăsându-i", needless to mention the negative counterparts like "nelăsându-mă", "nelăsându-te", "nelăsându-l", "nelăsând-o", "nelăsându-ne", "nelăsându-vă", "nelăsându-i" or even the forms with the adverb "mai" (En. any more) between the negative prefix and the verbal root: "nemailăsându-mă", "nemailăsându-te", "nemailăsându-l", "nemailăsând-o", "nemailăsându-ne", "nemailăsându-vă", "nemailăsându-i". Consequently, the hyphenated words lexicon was limited to the most frequent hyphenated words. For the rest of the hyphenated words occurring in the raw text the hyphen was replaced with space (see the left part of Fig. 1). This led to creating a LM comprising incomplete words such as "ul", "ului", "lăsându", etc., which eventually appeared in the ASR output text as errors. The solution to this problem was to implement a more complex hyphen processing procedure in the NLP application that preprocesses the raw text, before language modelling. The hyphen-words lexicon was kept, but only for compound words, such as "Târgu-Jiu". In addition to this lexicon, we created lists of hyphen prefixes[1] (*e.g.* "te-", "ne-", "istorico-") and suffixes (*e.g.* "-se", "-vă", "-ul"). Provided these resources, a hyphenated word occurring in the raw text was processed as follows:

if (hyphenated word is in the hyphenated words lexicon)	**Examples**
then keep its form intact	"Coca-Cola"
else split hyphenated word into parts	
if (one of the parts is in the usual word lexicon) && (the other part is in the prefixes/ suffixes lists)	"TVA" "-ul"
then the word and the prefix/suffix are separated by a space	"TVA-ul"=>"TVA -ul"
else the hyphen is replaced with a space and the initial hyphenated word is logged for further manual inspection	"două-trei" "bugetari-particulari"

Thus, after the text processing phase, the text used for the LM training comprised words such as: "să", "-mi", "să", "-i", "m-", "a", "ratat", "-o", "FMI", "-ul", "TVA". The new

[1] What we call *hyphen prefixes* and *suffixes* are clitics or definite articles that are attached to a word by means of a hyphen. Prefixes are attached to the front of the word, while suffixes to its end.

ASR system using this LM component now generates texts as follows: *"**TVA -ul** a crescut cu 3 la sută"*; *"**m- a** rugat **să -i** dau ceva"*. Finally, a simple text postprocessor binds the hyphens to the adjacent words (*e.g. "**TVA-ul** a crescut cu 3 la sută", "**m-a** rugat **să-i** dau ceva"*). The whole process is illustrated in Fig. 1.

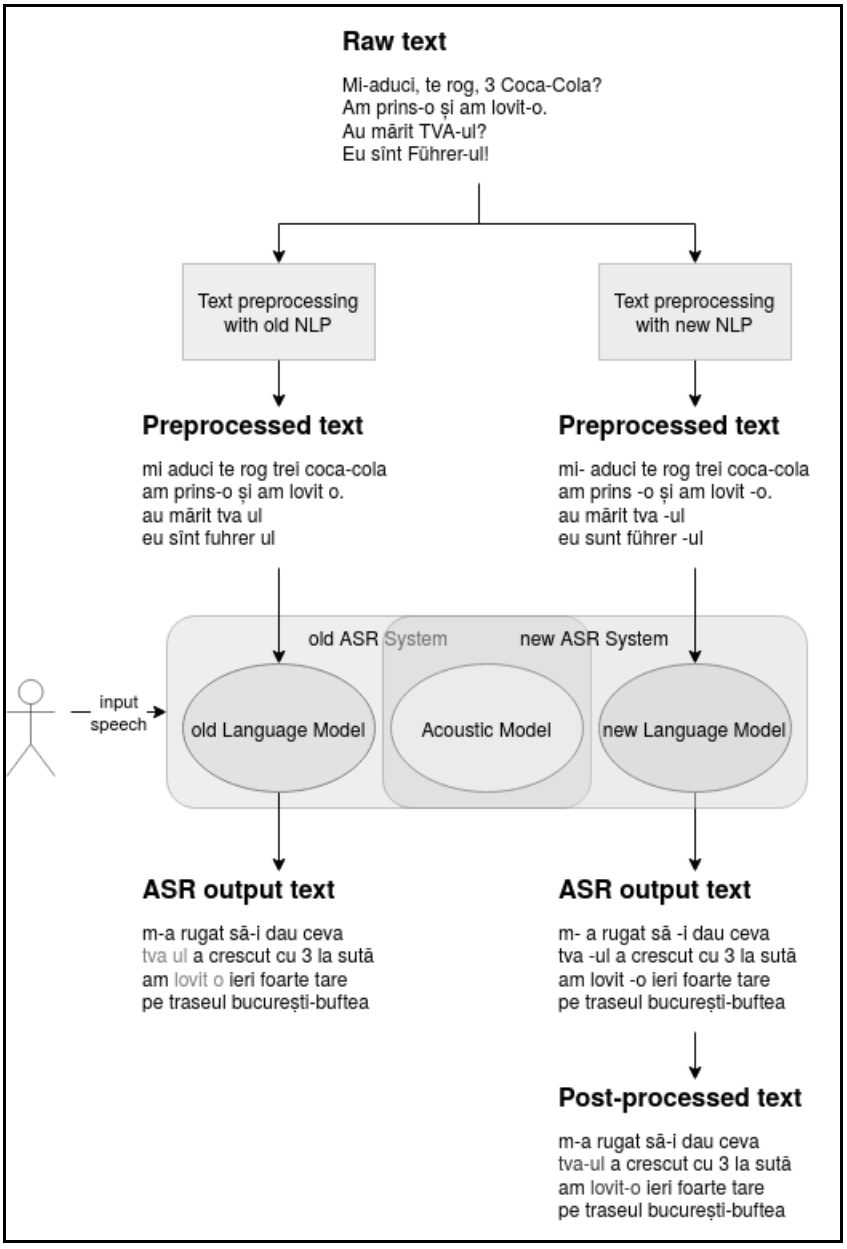


Figure 1: Different ways of processing raw text samples before training the language model (upper part of the figure) and various ASR output texts (lower part of the figure). Note that the ASR output is conditioned by the input speech, not by the raw text samples used for language modelling. However, the way in which the raw text samples are preprocessed before language modelling triggers specific errors in the ASR output.

3.3. Other updates of the natural language processor

The natural language processor had two more updates. One modification consists in the replacement of the character "î" with "â" inside words. This was due to the fact that some words were spelt according to the older norms, for example "cînd" instead of "când", but the application correctly omits the replacement for prefixed words such as "neînțeles", as well as for compounds such as "bineînțeles". This replacement process is done in two steps. First, the word is searched for in the lexicon; if it is found, it remains unchanged (*e.g.* "bineînțeles"), otherwise we move to step 2. In the second step, we replace the "î" character with the "â" character in the word as long as the character "î" is not positioned as the first or last character of the word (*e.g.* "cînd" is transformed into "când"); if the modified word is found in the lexicon, it will be kept in this form, otherwise it will be moved in a list of missing words for further manual inspection. As another minor update, special characters from other languages such as "ü" are no longer removed. These characters can be found in names and words and removing them would alter the name or word.

4. Experimental setup

This section presents the datasets used for training and evaluation and the baseline language models and ASR systems.

4.1. Datasets

Baseline LMs were trained on a plain text corpus, further called news002, comprising around 352 million words and collected by the SpeeD research group between 2008 and 2014 from five news websites. Improved LMs were trained on the union of the news002 corpus and the newly collected news2020 corpus. All LMs were evaluated on the transcripts of the following three speech datasets: RSC-eval[2] (Georgescu *et al.*, 2020), SSC-eval1[3] (Georgescu *et al.*, 2017) and SSC-eval2. SSC-eval1 and SSC-eval2 contain 3.5 hours, respectively 1.5 hours of spontaneous speech extracted from radio and TV broadcasts. RSC-eval comprises read speech and has been acquired in a laboratory environment, without background noise. ASR systems incorporating the baseline and improved LMs were evaluated on the speech part of the datasets mentioned above.

4.2. Performance metrics

We evaluate and compare the LMs in terms of out of vocabulary (OOV) rate and perplexity (PPL). The OOV rate is the percentage of words not available in the language model's vocabulary. The perplexity represents the prediction power of the LM: more

[2] RSC-eval used in this work is revision v2 of the RSC-eval dataset presented in (Georgescu *et al.*, 2020). Details regarding the differences between revision v1 used in (Georgescu *et al.*, 2020) and revision v2 used in this paper are provided in TADARAV 2020 report, section 1.1: http://tadarav.speed.pub.ro/storage/rapoarte/41.1._RST_in_extenso_TADARAV_2020_v3.pdf

[3] SSC-eval1 used in this work is revision v2 of the SSC-eval dataset presented in (Georgescu *et al.*, 2017). Details regarding the differences between revision v1 used in (Georgescu *et al.*, 2017) and revision v2 used in this paper are provided in TADARAV 2020 report, section 1.1: http://tadarav.speed.pub.ro/storage/rapoarte/41.1._RST_in_extenso_TADARAV_2020_v3.pdf

precisely, a high perplexity would suggest that the LM has a lower chance to estimate the sequence of words given as input, while a lower perplexity would mean that the LM expects the words to have that specific order. Thus, we could say that PPL represents the degree to which the given input sequences of words are common to the LM.

We evaluate and compare baseline and updated ASR systems in terms of the word error rate; *i.e.* the percentage of insertion, deletions and substitution errors relative to the number of words in the reference text.

4.3. Baseline language models

The baseline LM (Georgescu *et al.*, 2017) used for speech decoding in SpeeD ASR systems prior to this work was a statistical bigram model created using SRI-LM on the news002 corpus (~352M words). The model was created without n-gram pruning (it comprises all bigrams occurring in the training text) and its vocabulary was limited to the most frequent 200 words. This model is further called LM-2017-2g-large-200k.

The baseline language model used for language rescoring is the recurrent neural network (RNN) (Georgescu *et al.*, 2019) consisting of 3 time-delay (TDNN) blocks, where each block contains affine, ReLU and renorm layers. These blocks are interspersed with 2 long-short term memory (LSTM) layers. The context of this model is limited to the previous five words. This model is further called LM-2019-RNN.

The baseline LMs are evaluated and compared in Table 1. The rescoring LM is clearly better than the two-gram. However, it cannot be used in the speech decoding process due to computational reasons.

Table 1: Baseline language models. We report the results in terms of perplexity (PPL) and out-of-vocabulary (OOV) rate on the three evaluation datasets: RSC-eval, SSC-eval and SSC-eval2

	PPL [%]			**OOV [%]**		
Model name	**RSC-eval**	**SSC-eval1**	**SSC-eval2**	**RSC-eval**	**SSC-eval1**	**SSC-eval2**
LM-2017-2g-large-200k	316	219	356	0.13	1.96	0.91
LM-2019-RNN	176	132	325	0.01	0.82	0.66

4.4. Baseline ASR system

The ASR system used as a baseline in this paper is the state-of-the-art system for the Romanian language (Georgescu *et al.*, 2021). Its acoustic model is implemented using Kaldi (Povey *et al.*, 2011). The 220-dimensional input feature vector represents mel-frequency cepstral coefficients (MFCCs) extracted from 3 consecutive frames of 25 ms and iVectors extracted from chunks of 1500 ms. We use 6 factorized time-delay neural network (TDNN-F) blocks (Povey *et al.*, 2018), followed by two parallel output blocks on which the cross-entropy and lattice-free maximum mutual information (LF-MMI)

chain loss functions (Povey *et al.*, 2016) are applied. The acoustic model outputs 3760-dimensional posteriors for the acoustic states.

The baseline ASR system incorporates LM-2017-2g-large-200k for speech decoding and uses LM-2019-RNN for language rescoring. Its performance on the three evaluation datasets is presented in Table 2.

Table 2: Baseline ASR system performance (in terms of WER [%]) after speech decoding using LM-2017-2g-large-200k and language rescoring with LM-2019-RNN

ASR system	RSC-eval	SSC-eval1	SSC-eval2
Speech decoding using LM-2017-2g-large-200k	2.8	13.3	16.4
+ language rescoring using LM-2019-RNN	1.8	11.0	14.0

5. Experimental results

In this section we evaluate the improved LMs. Note that we do not introduce new LM types; we only train new LMs using an extended and better preprocessed text corpus.

The baseline LMs were trained only on the news002 corpus (352M words), while the improved ones were trained on the news002 corpus plus the news2020 corpus (352M words + 255M words). The text on which the improved LMs were trained is preprocessed using the new natural language processor discussed in Section 3.

The improved LMs are evaluated independently in terms of OOV and PPL and also incorporated in the ASR system, in terms of WER.

Simpler LMs (2-gram and 3-gram) were incorporated in the speech decoding phase of the ASR, while more complex LMs (4-gram and RNN) in the language rescoring phase. These latter models cannot be used in the speech decoding phase due to architecture and memory constraints.

5.1. Language models for speech decoding

For speech decoding we experimented with various n-gram orders (2-gram and 3-gram), various vocabulary sizes (200k, 250k and 300k words) and n-gram pruning (no pruning, 1e-7, 3e-7). The results obtained on the speech transcripts are provided in Table 3.

Table 3: Improved speech decoding language models (LM-2020-*) versus baseline model for speech decoding (LM-2017). The name of the models indicates the n-gram order (2g, 3g), whether n-gram pruning was applied or not (small = 3e-7, medium = 1e-7, large = no pruning) and the vocabulary size (200k, 250k, 300k words). We report the results in terms of perplexity (PPL) and out-of-vocabulary (OOV) rate on the three evaluation datasets: RSC-eval, SSC-eval and SSC-eval2

	PPL [%]			**OOV [%]**		
Model name	**RSC-eval**	**SSC-eval1**	**SSC-eval2**	**RSC-eval**	**SSC-eval1**	**SSC-eval2**
LM-2017-2g-large-200k	316	219	356	0.13	1.96	0.91
LM-2020-2g-large-200k	289	225	335	0.19	0.51	0.37
LM-2020-2g-small-200k	388	275	406	0.19	0.51	0.37
LM-2020-2g-small-250k	391	278	408	0.14	0.44	0.34
LM-2020-2g-small-300k	393	280	411	**0.09**	**0.37**	**0.28**
LM-2020-3g-large-200k	**178**	**131**	**210**	0.19	0.51	0.37
LM-2020-3g-medium-200k	243	167	259	0.19	0.51	0.37
LM-2020-3g-small-200k	294	195	299	0.19	0.51	0.37
LM-2020-3g-small-250k	296	197	300	0.14	0.44	0.34
LM-2020-3g-small-300k	298	199	303	0.09	0.37	0.28

In terms of PPL, the best LM, LM-2020-3g-large-200k, has a 40% relative decrease on the RSC-eval evaluation dataset, 44% relative decrease on SSC-eval1 and 41% relative decrease on SSC-eval2 when compared to the baseline system. In terms of OOV, the same vocabulary size models share the same results due to the fact that they have the same vocabulary. Compared to the baseline, the OOV results have seen an improvement on the spontaneous speech evaluation datasets, while on the read speech evaluation dataset, only the models with a vocabulary of 300k words have generated better results.

5.2. Language models for language rescoring

Similar to Section 5.1, where we evaluated the models dedicated to speech decoding, in this section we discuss the results obtained for the models designed for language rescoring, namely the 4-gram models and the RNN model: see Table 4. We can observe a relative decrease of 7% and 40% PPL on the RSC-eval and SSC-eval2 evaluation datasets for LM-2020-4g-large-200k when compared to the baseline model with

rescoring. For the SSC-eval1 evaluation dataset, LM-2020-5g-RNN generated better results with a 14% PPL relative decrease.

Table 4. Improved language rescoring models (LM-2020-*) versus baseline model for language rescoring (LM-2019). The name of the models indicates the model type (4g, RNN), n-gram pruning (large = no pruning) and the vocabulary size (200k, 250k, 300k words)

	PPL [%]			**OOV [%]**		
Model name	**RSC-eval**	**SSC-eval1**	**SSC-eval2**	**RSC-eval**	**SSC-eval1**	**SSC-eval2**
LM-2019-RNN	176	132	325	0.01	0.82	0.66
LM-2020-RNN	246	**113**	206	**0.01**	0.43	0.35
LM-2020-4g-large-200k	**162**	118	**196**	0.19	0.51	0.37
LM-2020-4g-large-250k	163	119	197	0.14	0.44	0.34
LM-2020-4g-large-300k	165	120	198	0.09	**0.37**	**0.28**

The best OOV results for the spontaneous speech evaluation datasets were obtained for LM-2020-4g-large-300k, while for the RSC-eval the best result is shared between the baseline model with rescoring and LM-2020-5g-RNN. The baseline model with rescoring and LM-2020-5g-RNN have an OOV of 0.01%. This low OOV value could be explained by the fact that these 2 models had no vocabulary size limit, whereas the 4-gram models were limited.

5.3. Improved ASR system

In this section we evaluate the ASR systems obtained by incorporating the LMs presented in sections 5.1 and 5.2. For speech decoding, we chose the 3-gram 200k models, while for language rescoring we opted for both LM-2020-4g-large-200k and LM-2020-5g-RNN. In Table 5 we present the WER results with and without language rescoring. The name convention for the language models is the one discussed in the previous sections.

We can notice in Table 5 that the best ASR system is the one which uses LM-2020-3g-large-200k with LM-2020-RNN for rescoring. This model has a relative decrease of 9% and 7% WER on the spontaneous speech evaluation datasets, namely, SSC-eval1 and SSC-eval2 respectively, when compared to the baseline ASR system with rescoring. However, for RSC-eval we can observe a relative increase of 16% WER.

Table 5: ASR WER results after decoding and after rescoring

LM used for language rescoring	LM used for speech decoding	WER[%] w/o rescoring			WER[%] w rescoring		
		RSC-eval	SSC-eval1	SSC-eval2	RSC-eval	SSC-eval1	SSC-eval2
LM-2017-RNN	LM-2017-2g-large-200k	2.8	13.3	16.4	1.8	11.0	14.0
LM-2020-4g-large-200k	LM-2020-3g-large-200k	2.7	11.0	14.4	2.8	10.7	14.1
	LM-2020-3g-medium-200k	3.1	11.9	15.1	2.7	10.8	14.0
	LM-2020-3g-small-200k	3.3	12.6	15.8	2.6	10.8	14.1
LM-2020-RNN	LM-2020-3g-large-200k	2.7	11.0	14.4	2.1	10.0	12.9
	LM-2020-3g-medium-200k	3.1	11.9	15.1	2.2	10.4	13.3
	LM-2020-3g-small-200k	3.3	12.6	15.8	2.2	10.5	13.7

5.4. Hyphenated words problem

The results from Section 5.3 have shown an improvement for the spontaneous speech evaluation datasets and a slight increase of the WER for the read speech evaluation dataset. In Table 6 we present a few examples of the hyphenated words problems, through a comparison between the outputs of the baseline ASR with language rescoring and the improved ASR system which uses LM-2020-3g-large-200k for decoding and LM-2020-RNN for rescoring.

Table 6: Comparison between old and new ASR outputs with regard to the hyphenated words problem. Correct hyphen word form is highlighted with bold.

ASR output with old text preprocessor	ASR output with new text preprocessor
[...] meciul cu pandurii **târgu-jiu**.	[...] meciul cu pandurii **târgu-jiu.**
[...] a continuat apoi adresându i câteva vorbe [...]	[...] a continuat apoi **adresându-i** câteva vorbe [...]
[...] a celei mai mari ţări sud americane.	[...] a celei mai mari ţări sud americane.

In the first row of Table 6, we can notice an example where both ASR systems have correctly generated the hyphenated word “târgu-jiu”, while on the second row an improvement has been made thanks to the fact that the word “adresându-i” contains a hyphen in the new ASR output, whereas that was not the case for the old ASR. Lastly, on row 3, no improvements have been made, both systems erroneously generated the word “sud americane” instead of “sud-americane”, which is found in the reference. This

is mostly due to the fact that the vocabulary used for the training contained words like "sud americane", as well as "sud-americane", which led the model to generate both versions in text. These are just a few examples of the NLP updates with regard to the hyphenated words problem. Unfortunately, as can be noticed for the WER results and examples presented in this section, these corrections have not occurred in most cases.

6. Conclusions

We presented our updates in improving our LMs used for the training of ASR systems. The main contributions of this paper are: (i) the acquisition of the new text corpus, news2020; (ii) updates brought to our natural language processor and (iii) new LMs for ASR systems. By adding the new text corpus and the different processing of hyphenated words we hoped to obtain better LMs in terms of PPL and OOV and consequently a better ASR system in terms of WER. The best LM presented slightly better results for the spontaneous speech evaluation datasets when compared to the baseline LM, while the ASR system trained with this model had a similar result in terms of WER, with a relative decrease of nearly 10%.

Acknowledgements

This work was supported by a grant of the Romanian Ministry of Research and Innovation, CCCDI – UEFISCDI, project number PN-III-P1-1.2-PCCDI-2017-0818 / 73PCCDI, within PNCDI III.

References

Cucu, H. (2011). Towards a speaker-independent, large-vocabulary continuous speech recognition system for Romanian, PhD Thesis, University "Politehnica" of Bucharest.

Cucu, H., Buzo, A., Besacier, L., Burileanu, C. (2014). SMT-based ASR Domain Adaptation Methods for Under-Resourced Languages: Application to Romanian. *Speech Communication Journal*, Vol. 56, 195-212.

Georgescu, A.-L., Cucu, H., and Burileanu, C. (2017). SpeeD's DNN Approach to Romanian Speech Recognition. In *Proceedings of the 9th Conference on Speech Technology and Human-Computer Dialogue* (SpeD), Bucharest, Romania, 1-8.

Georgescu, A.-L., Cucu, H., and Burileanu, C. (2019). Kaldi-based DNN architectures for speech recognition in Romanian. In *Proceedings of the 10th Conference on Speech Technology and Human-Computer Dialogue* (SpeD), Timișoara, Romania, 1-6.

Georgescu, A.-L., Cucu, H., Buzo, A., and Burileanu, C. (2020). RSC: A Romanian Read Speech Corpus for Automatic Speech Recognition. In *Proceedings of the 12th Language Resources and Evaluation Conference*, Marseille, France, 6606-6612.

Georgescu, A.-L., Manolache, C., Oneață, C., Cucu, H., and Burileanu, C. (2021). Data-filtering methods for self-training of automatic speech recognition systems. In *Proceedings of the IEEE Spoken Language Technology Workshop* (SLT), Virtual (in press).

Iordache, F., Georgescu, A.-L., Oneaţă, D., and Cucu, H. (2019). Romanian Automatic Diacritics Restoration Challenge. In *Proceedings of the 14th International Conference on Linguistics Resources and Tools for Natural Language Processing*, Cluj-Napoca, Romania, 65-74.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., and Vesely, K. (2011). The Kaldi Speech Recognition Toolkit. In IEEE 2011 *Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society.

Povey, D., Peddinti, V., Galvez, D., Ghahremani, P., Manohar, V., Na, X., Wang, Y., and Khudanpur, S. (2016). Purely sequence-trained neural networks for ASR based on lattice-free mmi. In *Proceedings of Interspeech*, 2751-2755.

Povey, D., Cheng, G., Wang, Y., Li, K., Xu, H., Yarmohammadi, M., and Khudanpur, S. (2018). Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Proceedings of Interspeech*, 3743-3747.

Tufiș, D., Barbu Mititelu, V., Irimia, E., Mitrofan, M., Ion, R., and Cioroiu, G. (2019). Making Pepper Understand and Respond in Romanian. In *22nd International Conference on Control Systems and Computer Science* (CSCS), 682-688.

CHAPTER 4.
APPLICATIONS

AUTHOR CONFIDENCE AS A PREDICTOR OF THE ACCEPTANCE OF SCIENTIFIC PAPERS

MIHAELA ONOFREI[1,2], DIANA TRANDABĂȚ[2]

[1] *Institute of Computer Science, Romanian Academy - Iasi branch, 700481 Iasi, Romania, mihaela.onofrei@iit.academiaromana-is.ro*
[2] *Faculty of Computer Science, "Alexandru Ioan Cuza University of Iași dtrandabat@info.uaic.ro*

Abstract

Research is a fundamental part of a country's innovation. Resources allocated to research and development around the globe have constantly increased over the last years, bringing along a higher number of articles submitted to scientific venues. In this context, we introduce a system which uses language technologies for predicting the evolution of the scientific life. More precisely, our system analyses submitted papers and extracts the self-confidence that authors demonstrate to have into their research. Using these author confidence scores, we further predict the acceptance chances for submitted papers. For this task, we collected a set of 260 scientific articles submitted to a scientific journal, along with their reviews. For each article, we identify author's confidence by automatically collecting a number of triggers from the submitted article, ranging from lexical and semantic up to reputation related features. Using this collection, we trained a Naïve Bayes classifier and evaluated its performance on a different set of 25 papers. The results show that our system correctly identified the assessment of the paper, as either accepted or rejected, with a precision of 0,60 and a recall of 0.75, which open the path for further investigations in this domain.

Key words — Academic Uses of Digital Collections, Author Confidence, Research Data Analysis

1. Introduction

Mining scientific literature to extract the science behind it, such as concepts, patterns or relations, is a very productive research area. However, extracting non-scientific information from scientific data has recently also seen an increasing interest, with applications ranging from identifying speculative language to retrieval of papers with a specific writing style, in an attempt to cope with different reading preferences.

This paper represents a proof of concept and it proposes a method to assess the degrees of confidence that an author has in his/her own scientific production intended to predict the likelihood of having the paper accepted at various conferences and journals. Experiments and results discussed in this paper are run over a set of data extracted from a semantic web journal, which also made available the reviews for the included papers. This survey is based on our previous research in identifying what features indicate the author's level of trust in his/her own scientific writing.

The paper is structured as follows: Section 2 briefly discusses recent approaches to author confidence and how language technologies can be used to evaluate the impact of

a scientific paper. Section 3 presents the architecture of our system, designed to identify important features for evaluating author's confidence and assess a scientific paper's acceptance chances. Section 4 describes each module of the system, at lexical, semantic and reputation levels. The scores each module assigns to the paper are combined through machine learning to predict how publishable the paper is. The results are presented in Section 5, and the final section reveals further challenges.

2. Background

Over the last years, a significant growing amount of scientific work is noticed (Larsen and von Ins, 2010), and the retrieval of relevant literature represents a significant challenge for Ph.D. students and scientific researchers, for both finding the latest breakthrough or for compiling a state-of-the-art for an area of interest. Kreiman and Maunsell (2011) state that the evaluation of scientific productivity is a thorny problem. Traditionally, scientific work is evaluated using peer review in the form of evaluation from expert reviewers that judge the value, rigor of new findings. This type of evaluation plays a critical role, but its subjective nature limits it.

These limitations have motivated scientists to explore more quantitative measures that can increase the evaluation by peers. Such examples range from the number of publications as a metric of evaluation (Refinetti, 2011), journal impact factors (Dimitrov *et al.*, 2010), h-factor (Hirsch, 2005), page ranks, article download statistics to comments using social media (Mandavilli, 2011). To complement peer-review efforts, Kreiman and Maunsell (2011) propose nine quantitative criteria for scientific contributions of individuals within a field. In recent years, there has been an increasing interest in how an author's credibility influences paper acceptance. For instance, Aguinis *et al.* (2019) found that insufficient details about data collection and preparation procedures could be explained by authors' lack of knowledge or lack of confidence, which indicate that the researchers are not transparent, thus not credible.

For identifying the author confidence, we tested different types of Sentiment Analysis tools. Even if sentiment analysis is receiving increasing interest (Cieliebak *et al.*, 2013), its accuracy on scientific documents needs significant improvements. In a recent paper, Hussein (2016) proposes a new technique to evaluate sentiment analysis for online scientific papers. This technique relies on topic parameters and sentiment analysis on existing reviews. In order to improve accuracy, Hussein (2016) presents an enhanced Bag-Of-Words model that depends on a word weight, instead of the term frequency of each word. Besides, the system evaluates a score that relies on topic domain parameters (place of publication, number of citations, and publishing paper date), bi-polar sentiments, implicit and explicit negative, and world knowledge.

The inspiration for our research was the study by Cyra and Gorski (2007), investigating the relation between an individual's self-reported confidence and the influence they had within a freely interacting group. They concluded that the influence of an individual within a group was directly dependent on his or her confidence level.

In this context, we tested if a confident scientific paper will have increased chances of being selected, either for reading or for approval in various scientific journals, as compared to a similar paper, but written in a less confident manner. Therefore, we

developed an instrument for identifying an author's confidence, based on writing style, reputation and the sentiment revealed in the analysed article.

3. Architecture

While the study of the connection between discourse patterns and personal identification of an author is decades old, the study of these patterns, using language technologies, is relatively new. In the more recent tradition, we frame author's confidence prediction from a text as an important problem for the natural language processing domain. Confidence (Light *et al.*, 2004) is generally described as a state of being certain either that a hypothesis or prediction is correct or that a chosen course of action is the best or most effective. Different approaches consider the confidence in terms of "appropriateness" or "trustworthiness" (Derntl, 2019), or correlate it to uncertainty. In (Light *et al.*, 2004) the authors describe a function theory, called Dempster-Shafer (D-S), for evaluating the confidence of an argumentation. In (Wang *et al.*, 2016), a trust case framework is used to check the argumentation used to demonstrate the compliance with specific standards.

In the context of this study, a structured argumentation, although it plays an important role in the communication, is not enough. Automatically discovering if an author is confident or not in his argumentation is a challenging task, which involves finding author's sentiments, features to determine his writing style, as well as information about his mastering of the scientific field.

Figure 1. Architecture of our author confidence system

In order to determine the confidence of an author in his work, we propose a system composed of two main modules (see Figure 1): a scorer to analyse each article and extract relevant features towards the identification of author confidence, and a machine learning module used to predict the article's acceptance. In a previous study (Onofrei *et al.*, 2019), we found out that the Results and Conclusion sections usually contain a higher degree of subjectivity and are therefore more likely to reveal a confident or reluctant manner of presentation.
The architecture of our paper assessment system will be detailed in the next section, where each module is documented.

4. System description

4.1. Crawler

As depicted in Fig. 1, our system is composed of three main modules. The first one is a crawler allowing the creation of a working corpus. Using the crawler, we managed to collect a set of articles from the Journal Semantic Web – Interoperability, Usability,

Applicability[1], known as the Semantic Web Journal (SWJ), published by IOS Press. The journal is a forum where researchers from different domains submit papers which share the vision and need for more effective and meaningful ways to share information across agents and services on the future internet and elsewhere.

SWJ contains over 900 English documents from 2010 to date. One of the most important features that determined us to choose this journal was the existence of open reviews. In addition, the submitted manuscripts are posted on the journal's website and are publicly available. We could thus analyse the content of an article before and after the review process made by the experts in the field. This type of journal, which stores scientific papers together with their reviews, makes it easier for scientists to search for information and has been proven powerful resources in many data mining, machine learning, and information retrieval that require high-quality data (Caragea *et al.*, 2010).

The reason for selecting a subdomain (Semantic Web), instead of the broader Computer Science domain, was the use of the feature analysing specialized language. We expect papers from other CS domains to be comparable in terms of specific used vocabulary.

From the total of over 900 papers available on the Semantic Web journal website, we collected for the proof of concept of our system 260 papers along with their reviews and decision. The selected papers had been submitted to the journal (95%) or to the EKAW conference (5%), also hosted on the journal section. The possible decisions for the papers are: acceptance, rejection or revisions needed, either minor or major, see Table 1 for their distribution. Some accepted papers had also previous versions, where revisions were requested. They were used for the evaluation of our system (see Section 5).

Table 1: Distribution of scientific articles from the Sematic Web Journal platform in our corpus.

	Journal	**Conference**	**Total**
Accepted papers	58.46	1.92	60.38
Rejected papers	24.62	3.08	27.69
Revisions requested	11.92	0.00	11.92

The crawler extracts the PDF versions of the articles, along with a set of metadata, describing the title, the authors and the decision after review. In order to process the articles collected by the crawler, we used an online Optical Character Recognition software[2] and converted them into a collection of 260 TXT files. The next step involved a pre-processing of the TXT files in order to tokenize and lemmatize each word in the article, which are then fed to the author confidence detection system.

4.2. Scorer

The second module aims to assign a score for each group of features which we identified as relevant for the identification of author confidence.

[1] http://www.semantic-web-journal.net/
[2] https://www.onlineocr.net

Among these features, we focused on the lexical, semantic and credibility related ones, thus developing three scientific paper analysers.

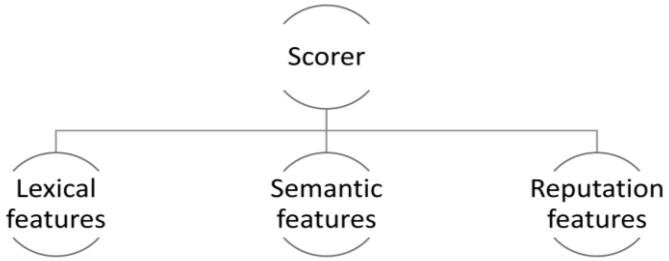


Figure 2: The structure of the module responsible with assigning confidence scores to scientific papers

Lexical

A first level of investigation is related to the words chosen by the authors to present their work, analysed from a qualitative and a quantitative point of view in order to assess the paper's readability. For this analysis, we only used sentences from the Results and Conclusion sections, as explained above.

There are many readability metrics in the literature, estimating the educational grade level necessary to understand a document. The FOG readability metric, revised in (Audisio *et al.*, 2009), computes paper readability based on the number of monosyllabic vs. complex words (*i.e.* words with more than 3 syllables, suffixes excluded). Our approach is similar, using variables such as the average words per sentence ratio and the frequency of specialized terms.

After tokenizing the text of each paper and identifying words, their lemmatized form is needed to compute accurate frequencies. Thus, unique unigrams and bigrams frequencies are computed and normalized by the length of the document and the number of tokens within. Functional words are removed and the proportion of specialized terms in each document is computed.

Although specialized jargon needs to be used to prove mastery of a domain, if the percent of specialized words is too high it decreases the readability of the paper. For the identification of the specialized language ratio, we used a glossary of around 280 expressions relevant for the semantic web domain, collected from online resources.

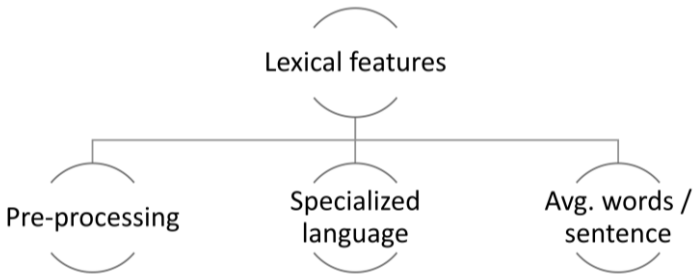


Figure 3: The analyzer for the lexical features

Inspired by the FOG metric, the final score for the lexical level of analysis is given by the following formula:

$$S(lexic) = 0{,}15 * \frac{\sum_{i=1}^{n} WPS_i}{\#sentences} + 0{,}25 * Specialized\_language$$

where WPS_i is the word per sentence ratio for each sentence in the Results and conclusion part of the paper and n iterated over all the sentences. The value for the word per sentence ratio is computed using the equation:

$$WPS_i \begin{cases} \frac{words_i}{sentence_i}, \text{if} 10 \leq \frac{words_i}{sentence_i} \leq 25 \\ 0, \text{otherwise} \end{cases}$$

According to Oxford Academy[3], it is highly recommended to use up to 15 words in a sentence, as higher rates of word per sentence affect readability. Since we are analysing scientific articles, and not regular texts, we extended this range to 25 words per sentence. We also introduced an inferior limit, since short sentences are relatively rare in the presentation of research.

For the specialized language, we used the equation below, with the threshold obtained in our previous research (Onofrei *et al.*, 2019):

$$Specialized\_language = \begin{cases} \frac{\#specialized\_words}{\#words} * 3, \text{if} < 0.33 \\ 0.25, \text{otherwise} \end{cases}$$

Semantic

The second module analyses the article at a semantic level, trying to identify the overall sentiment of the paper. While sentiment extraction is a topic where substantial research was recorded in the final decade, researchers have mostly focused on social media or shopper's reviews. For scientific papers, this research direction is yet at its beginning. It may at first seem legitimate, since a scientific paper is, by definition, neutral, and can have no sentiment attached. Even if the expressed sentiments differ from the ones expressed in social media or travel reviews (where emotions range from happiness or joy to sadness or anger), scientific papers can also have a form of sentiment.

By definition, sentiment analysis refers to the use of natural language processing to identify subjective information. Signalling the results presented in a subjective manner in a scientific paper is thus exactly in the scope of sentiment mining. In the biomedical domain, this direction has been established under the name of hedge detection (*i.e.*, speculative and tentative statements) (Light *et al.*, 2004).

Sentiment analysis of citations in scientific articles (*i.e.* if the author presents in a bright or darker light the cited article) is a new and interesting research topic that can open up many directions in bibliometrics. Athar and Teufel (2012), for instance, have created a corpus for their research focused on context-enhanced citation sentiment detection.

[3] https://www.ox.ac.uk/sites/files/oxford/field/field_document/Tutorial%20essays%20for%20science%20subjects.pdf

Another direction in identifying the sentiment in scientific papers is presented by Hussein (2016), where sentiment refers to the strength of opination and ambiguous bi-polar triggers. For instance, although the word “old” usually has a negative connotation (for example in “this is an old debate”), it can also have a positive sentiment if used in constructions such as "statistic approaches have an old tradition in the field".

Therefore, we believe sentiment extraction can be beneficial for the analysis of scientific papers, as the author’s self-confidence influences the way a paper is written.

We used for our research the Stanford Sentiment Analysis tool[4]. Their deep learning model builds up a representation of a whole sentence based on its grammatical structure. Stanford Sentiment Analysis tool computes the sentiment based on how words compose the meaning of longer phrases, using a Recurrent Neural Network. The sentiment is expressed as polarity (*i.e.* the text tends to be positive or negative). After analysing each sentence individually, a score for the entire document is computed.

Reputation

This analyser extracts information for the author’s reputation profile (see Fig. 4). The importance of a paper for its domain is estimated computing the author’s notoriety and investigating his/her publication record using Google Scholar. The most important factors which enter in the construction of our score for reputation are the number of publications, the number of times the author was cited and the number of citations that the author’s papers received over time.

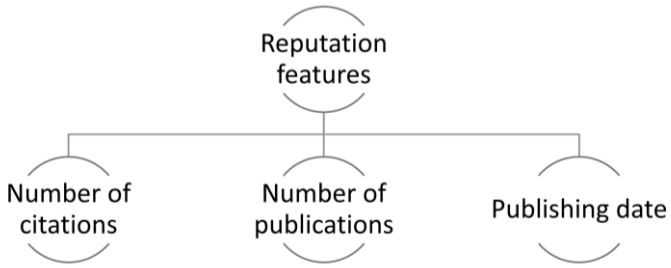


Figure 4: The features used by the reputation analyser

The citation score for each paper also considers how long ago the paper was published, using the formula:

$$S(citation_i) = \frac{\#citations_i}{currrent\_year\text{-}publication\_year_i}$$

where $\#citations_i$ is the total number of citation for paper i. The bigger the distance between the publication date and the current year, the higher the probability of having a bigger number of citations if the author published relevant research.

The final reputation score for the authors is computed using the formula:

$$S(reputation) = 0{,}15 * \frac{\#citations}{1000} + 0{,}05 * \frac{\#publications}{10} + S(citation)$$

[4] https://nlp.stanford.edu/sentiment/

The weights were empirically identified, using information from the corpus, but also from various online good practice guides on how to write a scientific article.

For a further development of this module, we intend to also consider the rank of the journal/conference where author's papers are published, as well as those of the paper which cited the author, as an indication of the quality of an author's work.

Each of the three main analysers (lexical, syntactic and semantic) returned a score for each article, and the next module takes these scores and trains a Naïve Bayes classifier to predict the acceptance of a paper.

4.3. Classifier

Using the scores from the previous three analysers, we created an instance for each article and trained a Naïve Bayes classifier. We considered the papers where reviewers requested minor or major revisions as rejected, so the distribution of the data was 60% accepted papers, 40% rejected ones. The classifier was trained on the 260 documents in our corpus, and tested on a newly extracted set of 25 documents. We kept the 60-40 distribution of the articles also in the test data. The results are discussed in the next section.

5. Results

Our system aiming at predicting the acceptance/rejection of scientific papers scored a 0,64 accuracy. The results are presented in the form of a confusion matrix in table 2.

Table 2: Confusion matrix for the author confidence system

	gold accepted	**gold rejected**	**total**
System accepted	36%	12%	48%
System rejected	24%	28%	52%

The precision is 0.60, the recall 0,74, and the F-measure is 0.67. We believe that this is an encouraging result. When analysing the results, we noticed that the lexical analyser had variable influence in selected documents. However, the sentiment analyser needs improvements since its score introduced the biggest error rate. A general sentiment analyser is not entirely relevant when applied to scientific texts. For a further step, we intend to include more refined methods in the sentiment score, including the tone of the author, related to its own results (subjectivity), but also to the state of the art (how (s)he relates to the cited references). As expected, the module with the biggest impact was the reputation analyser. When we ignore its score, the recognition rate drops by 27%.

Another observation is that we treated articles with minor or major revisions alike, *i.e.* as rejected. However, some of the articles with minor revisions we included in our training corpus were sent a few weeks later to the journal and accepted, the accepted version being also included in the corpus. Therefore, we need to check the corpus to

include only one version of the paper, either the accepted or the revision one, since the same author writing style and reputation confuses the system.

Another drawback we noticed during our tests, and which needs to be improved in the future, is that the runtime of our system is about 2 minutes per paper. This is due to the fact that the reputation analyser sends repeated queries over the internet for each author and co-author of the paper.

6. Discussions

In this paper, we have presented a method for automatic evaluation of a scientific paper's acceptance or rejection based on author confidence. We show that, in order to automatically predict the author's level of confidence in his/her own scientific writing, we have studied the relation between lexical analysis (frequencies of specialized language, word per sentence ratio), semantic features (sentiment analysis), and author reputation profiling. Each of these three levels of analysis computed a score for the article and the scores were further used by a Naïve Bayes classifier to predict the paper's assessment. Our approach benefits from combining lexical analysis, semantic analysis, reputation analysis with machine learning techniques and the experiments presented in this study show that this approach is highly promising to predict author's level of confidence in his/her own writing. To improve the performance of our system, we intend to enrich the gold annotated corpus with articles from different open review journals, as well as use additional machine learning techniques for the classification. As further work, the first direction will be to improve the sentiment analyser and adapt it for scientific texts. Additionally, language typos or errors can be addressed using tools similar to Grammarly[5]. We also have a set of features we want to test in the future, such as the ratio of text/image or formula or the use of empirical claims, such as cliché.

Acknowledgements

This work was supported in part by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS/CCCDI – UEFSCDI, project number PN-III-P1-1.1-PD-2019-0660.

References

Aguinis, H., Hill, S.N., Bailey, J.R. (2019). Best Practices in Data Collection and Preparation: Recommendations for Reviewers, Editors, and Authors. *Sage Journals*, https://doi.org/10.1177/1094428119836485.

Athar, A., and Teufel, S. (2012). Context-enhanced citation sentiment detection. In *Proc. of NAACL 2012: Human language technologies*, Association for Computational Linguistics, 597-601.

Audisio, R.A., Stahel, R.A., Aapro, M.S., Costa, A., Pandey, M., Pavlidis, N. (2009). Successful publishing: How to get your paper accepted. *Surgical Oncology*, 18(4), 350-356, doi:10.1016/j.suronc.2008.09.001.

[5] https://www.grammarly.com/

Caragea, C., Wu, J., Williams, K., Khabsa, M., Teregowda, P., Gilles, L.C. (2014). Automatic Identification of Research Articles from Crawled Documents. In *Proceedings of WSDM – WSCBD*.

Cieliebak, M., Dürr, O., & Uzdilli, F. (2013, December). Potential and Limitations of Commercial Sentiment Detection Tools. In *ESSEM@ AI* IA*, 47-58.

Cyra, L., Gorski, J. (2007). Supporting Compliance with Security Standards by Trust Case Templates. In *Proc. of the 2nd International Conference on Dependability of Computer Systems (DepCoS-RELCOMEX 2007)*, Poland, 14-16 June 2007, 91-98.

Derntl, M. (2019). Basics of Research Paper Writing and Publishing. *J. Technol. Enhan. Learn*. 6, 105-123.

Dimitrov, J. D., Kaveri, S. V., & Bayry, J. (2010). Metrics: journal's impact factor skewed by a single paper. *Nature*, 466(7303), 179-179.

Hirsch, J.E. (2005). An index to quantify an individual's scientific research output. In *Proceedings of the National academy of Sciences*, 102(46), 16569-16572.

Hussein, D.M. (2016) Analyzing Scientific Papers Based on Sentiment Analysis (First Draft), In: MsC thesis for Cairo university, https://www.researchgate.net/publication/301649777.

Kreiman, G., Maunsell, J. (2011). Nine Criteria for a Measure of Scientific Output. *Frontiers in Computational Neuroscience*, 5:48.

Larsen, P.O., von Ins, M. (2010). The rate of growth in scientific publication and the decline in coverage provided by Science Citation Index. *Scientometrics*, 84(3), 575-603.

Light, M., Qiu, X.Y., and Srinivasan, P. (2004). The Language of Bioscience: Facts, Speculations, and Statements in Between. In *Proc. of the BioLINK 2004: Linking Biological Literature, Ontologies and Databases*, Boston, MA, USA, 17-34.

Mandavilli, A. (2011). Peer review: trial by Twitter. *Nature* 469, 286-287.

Onofrei, M., Trandabăt, D., Gîfu, D. (2019). Towards Identifying Author Confidence in Biomedical Articles. *Data*, Volume 4, 18.

Refinetti, R. (2011). Publish and flourish. *Science*, 331(6013), 29-29.

Wang, R., Guiochet, J., Motet, G., and Schön, W. (2016). D-S Theory for Argument Confidence Assessment. In *Proc. of the 4th International Conference on Belief Functions (BELIEF 2016)*, Prague, Czech Republic, 21-23 September, 190-200.

ACCESSIBILITY SOLUTION FOR POOR SIGHTED PEOPLE AND ELDERLY AS AN END-TO-END SERVICE FOR APPLICATIONS. ROMANIAN APPROACH

CAMELIA-MARIA MILUȚ, ADRIAN IFTENE

Faculty of Computer Science "Alexandru Ioan Cuza" University of Iasi

camelia.i.milut@info.uaic.ro, adiftene@info.uaic.ro

Abstract

Through mobile and web applications, any information we need is at one click distance. This is not the case for people without the gift of sight or too little of it, who are often not considered by developers because they are a minority. Many times, adding special features for users with special needs can be an expensive part of a project, from a development time point of view. This paper presents a prototype of a generic accessibility solution that can be easily integrated into an already existing application in order to offer a more inclusive environment for all users. The solution is composed of two parts: a front-end and a back-end component. The front-end component is exposed as a library and mainly executes client-side processing, as speech to text. The back-end component is a serverless cloud native solution having a microservices-oriented architecture making it open to extension. The two components can be used together or separately, depending on the application's needs.

Key words —serverless, speech to text, visually impaired people.

1. Introduction

According to the World Health Organization, blindness and vision impairment affect at least 2.2 billion people around the world, the majority being over the age of fifty[1]. Reduced or absent eyesight can have major and long-lasting effects on daily personal activities, interacting with the community, school and work opportunities, and the ability to access public services.

In current times, many companies are trying to develop their products to be inclusive and accessible. A well-known example is Apple's VoiceOver, a screen reader for their products, and Amazon's Alexa, with an entire platform for distributing and developing voice-based applications (Miluț *et al.*, 2019; Filimon *et al.*, 2019). Other examples of accessible technology are smart assistants, which are applications that help the user in simple tasks like accessing the phone functionalities or finding the latest news (Calancea *et al.*, 2019). The question that this paper is trying to answer is *"How can we make already existing applications accessible for people with disabilities in an easy and rapid way for the developers?"*.

The paper is further structured as follows: Section 2 presents the solution proposed by us at the level of flow and architecture, at the level of front-end layer and at the level of

[1] https://www.who.int/health-topics/blindness-and-vision-loss

back-end layer. Section 3 presents the evaluation of the proposed solution and various scenarios for using the application. The paper ends with the sections of conclusions and bibliography.

2. Proposed Solution

There are several use cases identified as common in many applications and found to be troublesome to use by people with reduced eye capacities or the elderly. The *first use case* and most obvious feature that can be easily observed in many applications is that text is used as the main output format. Text can be difficult to read especially with intricate or small fonts and a variety of languages. The *second use case* regards information hidden in images. A concluding example here would be promotional banners from e-commerce sites that can include information about the latest offers in a banner that only contains an image. In this case, even if an application has a text to speech feature, the text from the image will be passed by. Another representative feature for inaccessible applications is hard to reach page elements like fields in a form. Fields in a form may be structured in such a way that they are hard to distinguish or so close together that they are hard to reach individually. A similar issue is faced with applications for content writing, like notes or emails (Kirkpatrick *et al.*, 2018). Using a keypad may also be difficult, particularly the touch screen keypads.

In this section, we will get into the technical details of a solution that answers the needs presented earlier.

2.1. Flow and Architecture

The solution is composed of two layers: the front-end layer and the back-end layer, as presented in Fig. 1.

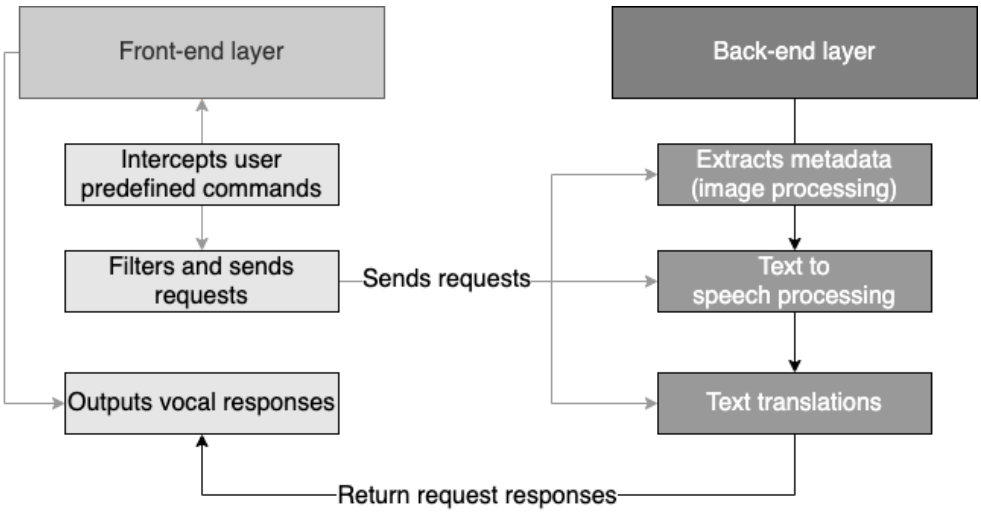


Figure 1: Proposed architecture

The front-end side can be used as a library that can be integrated with the front-end side of the host application. The main functionalities of the front-end layer are: (1) *intercepting user commands*, (2) *filter and process the commands from the user,* and (3)

forward a request in order to output a vocal answer. The back-end layer is built on cloud infrastructure as a collection of microservices and exposed through an API endpoint (Heymann *et al.*, 2001; Baboi *et al.*, 2019). Having the API endpoint, it can be further used by the front-end layer, or other use cases defined by the end-user (the developer). The main responsibility of this component is to process and delegate the client requests to the corresponding microservice. After choosing the corresponding functionality, the back-end layer fulfils its purpose by extracting (image metadata) and transforming (text to speech, translations) the necessary information.

2.1.1 Front-End Layer

The front-end layer handles the direct user integration, exposing the back-end functionalities plus client-side features. The client-side features can be easily described as *capturing user input* (spoken commands or text) and *output responses* (speech or text) (see Fig. 2).

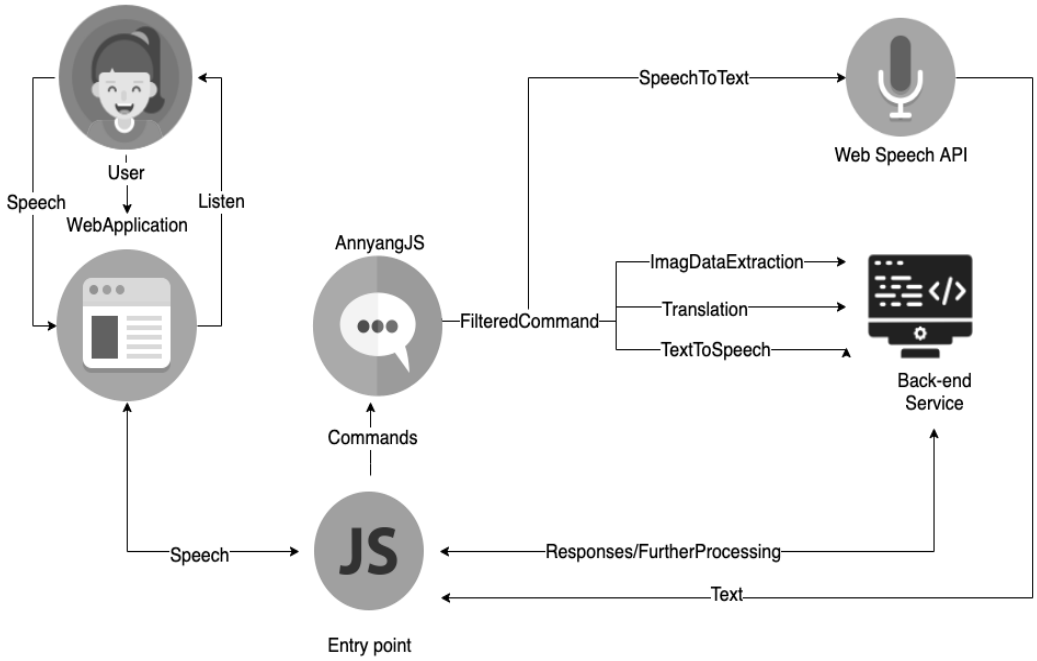


Figure 2: Front-end layer

Annyang API[2] was used to capture user commands. The front-end layer also uses WebSpeech API (Adorf, 2013) for larger amounts of speech that is about to be processed into text. Another available feature of this API is speech synthesis, also known as text to speech; unfortunately, it is not well trained for Romanian so it was not considered for this solution.

Fig. 2 illustrates how the structure of the front-end service and the subcomponent flow. The user interacts with the application using vocal commands or in the traditional way, using links and buttons, depending on the developer's implementation and, of course, the user's physical limitations. The application will be capable of delivering responses

[2] https://github.com/TalAter/annyang

to the user in such a way they would be easier to understand: text extracted from images and played out loud, text dictated and played out loud, text translated to Romanian from a various range of foreign languages (the service being able to detect from 55 languages and variants).

At the implementation level, the front-end layer contains JavaScript scripts, which are easy to integrate with most web applications as dependencies. The scripts define entry points for the service functionalities from the host application. Adding these functionalities over an application, the user will be able to: navigate through a menu using voice, complete a form using voice, have contents of an image described to them, obtain translations into Romanian, create notes and written content using speech, and also receive audio feedback (verify the content of a dictated note, receive the processed content as an audio output and other use cases that can be defined by the developer).

2.1.2 Back-End Layer

The back-end component is presented in two forms: (1) a ready to use API and (2) an application building block through the infrastructure as code capabilities (Amazon Web Services, 2017).

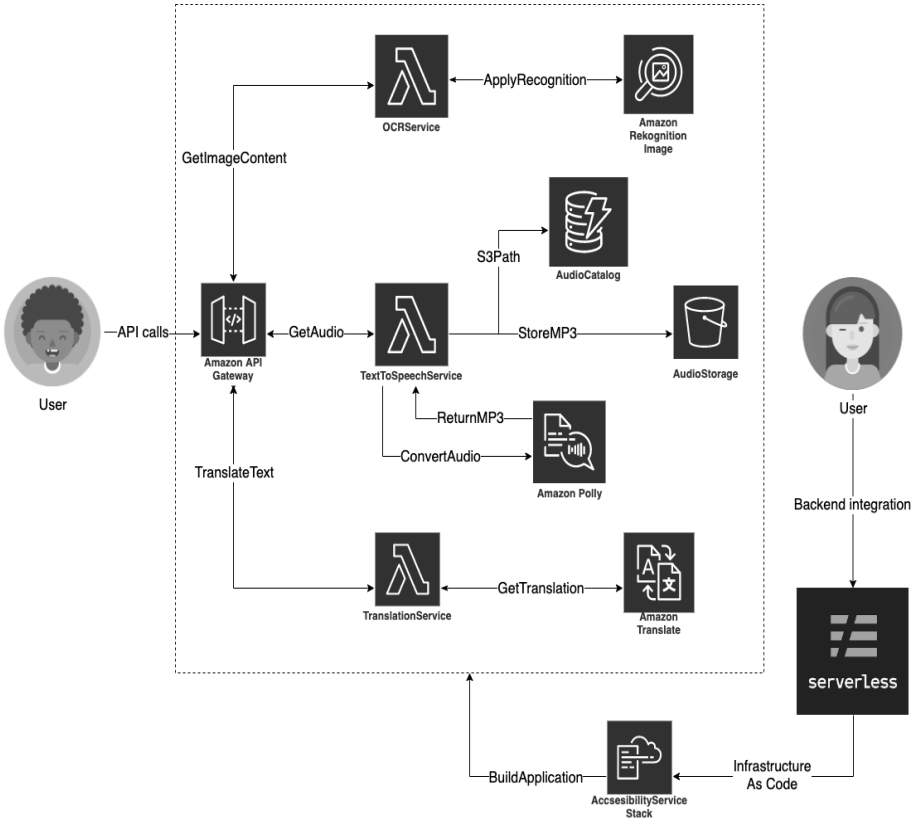


Figure 3: Back-End Layer

An application building block can be defined as a component that can be integrated into the code base and infrastructure of an already existing application. At the development level, the back-end layer is presented as a collection of microservices, accessible

through an API endpoint. The infrastructure is hosted on the cloud environment offered by Amazon Web Services[3].

Fig. 3 represents the cloud services (Amazon, 2020) involved in building this component, the interaction between them and the access points for the user. The flow of the service is the following: the entry point is represented by the API endpoint, which uses API Gateway under the hood. API Gateway is connected to the Lambda functions that implement the core functionalities and exposes them as a RESTful Web Service (Malakhov *et al.*, 2018). Customer requests are filtered by the API Gateway which delegates the request to the corresponding Lambda function. The lambda functions implement the following functionalities: image processing, text to speech, and translation.

ImageProcessingService uses Rekognition service[4] in order to extract the image metadata and return the text detected. As an input, the function receives a link to an image and returns the text detected.

The next function, TextToSpeechService, handles text to speech: it receives text as an input and outputs the link to an mp3 file stored in Simple Storage Service (S3) containing the voice translation. The input text is transformed into a voice using AWS Polly[5] (a service that provides already trained models for a variety of languages, including Romanian), resulting in batches of an audio stream, that is afterward translated into an mp3 file that will be further stored in S3. Files stored in S3 have a retention policy of 30 days, to be compliant with the data retention policy imposed by the European Union.

The third function is responsible with translating text into Romanian. TranslationService receives a text as an input, detects the origin language and outputs the translation into Romanian. The translation service is neural network based, meaning that it is able to detect the context of the source and provide a more fluent translation than the traditional rule-based translation services.

The infrastructure described in the previous paragraph is deployed through a CloudFormation template[6] (formed using the Serverless deployment framework[7]), that contains all the necessary configuration in order to have all the components ready to use in the AWS account of the developer and easy to modify using the developer command line. A customer can use this functionality as a component of an application, using storage hosted in their own accounts, being easier to handle issues such as customer privacy and GDPR[8].

[3] https://aws.amazon.com/free/
[4] https://aws.amazon.com/rekognition/
[5] https://aws.amazon.com/polly/
[6] https://aws.amazon.com/cloudformation/resources/templates/
[7] https://aws.amazon.com/free/
[8] https://gdpr-info.eu/

3. Evaluation

Evaluation for the solution consisted of two steps. The first one meant creating a host application that can be integrated with the front-end and back-end functionalities, in order to see how the integration can be done.

The second step was to give the enhanced application to a group of final users that can use the application and provide feedback. In order to test the functionalities, a skeleton application was developed. The application acted as a canvas for the front-end and backend functionalities, containing a homepage with a menu and pages for completing a form, describing an image content, creating notes by dictation, and translating the text into the Romanian from different languages. The interface is created in Romanian language and the vocal interaction is also performed in Romanian. In order to make the application available, it was deployed in AWS using Simple Storage Service (S3) hosting features[9].

3.1 Application Scenarios

3.1.1 Home Page Voice Controlled Menu

The first page of the host application contains a table of content that directs the users to the functional pages. In order to access a page from the menu, a user can use the command "Mergi la *" (En: "Go to *") where "*" is the placeholder for the destination page.

3.1.2 Complete a Form

The form contained by the application has the basic fields needed to complete an online purchase. Proposed fields are *first name, last name, user name, email, address* (street, city, country, and postal code), and a checkbox used to retain the data for the next purchase. A user interacts with this page as follows:

1. When the page is open, it asks for the permission to use the microphone, unless already allowed.
2. After the permissions are received, the page is ready to intercept the vocal commands.
3. A form is usually composed of fields identified by labels.
4. Following this rule, a command used over a form will identify a label and place the user on the corresponding field that is about to be completed.
5. The portion of the speech that is not identified as a label will be used to complete the action requested by that specific field.

3.1.3 Image Description

The image description page is composed of an image containing text and two buttons: one for obtaining only the text from the image, and a second one that gives the vocal

[9] https://aws.amazon.com/s3/

output for the detected text. This page integrates two features of the backend service: image processing and text to speech. Once the option for listening is selected, the text is immediately processed and played on the page without the need for additional user intervention.

3.1.4 Create Dictation Notes

Create dictation notes page allows the user to create email contents, personal notes, or any type of written communication using their voice instead of a keyboard. After the content creation, the user will be able to hear back what was written. This page represents a combination of the front-end service functionality of text to speech and the backend functionality of speech to text.

3.1.5 Translate into Romanian

Translating into Romanian is a feature specially designed for those who are more reluctant to technology because of the language barriers that can occur from time to time. In the test application, the translation feature is presented on a page containing text in five different languages; from the fifty-five available languages and variants, the one presented here are: English, Spanish, French, Dutch and Italian.

The texts are presented in different sections with a button at the end that triggers the translation. The backend service automatically detects the source language and translates the text into Romanian. The translation is displayed at the end of the section as text, as a demonstration for this service only.

3.2 Usability Testing

The application described in the previous section was put to test by a group of end-users. The group of users rated the application using the System Usability Scale (SUS) (Brooke, 1986).

The group of users that participated in this experiment was composed of thirteen people from which six had troubled vision, with age ranges from eighteen to fifty-two, with technical and non-technical backgrounds, with good and bad vision, simulating a real-world audience for an application. The scope of the experiment was to explore how easy to use is an application designed for inclusiveness.

Each volunteer followed a testing scenario composed of the following steps:

1. *Home page*: use the vocal command menu to reach the other pages;
2. *Form page:* complete the presented fields using voice;
3. *Describe image page*: obtain image content as text, translate the obtained text to speech;
4. *Dictation page*: start dictation and start speaking, see the spoken text in the text area of the page, listen to the resulting input, use the text further;
5. *Translation page*: get a translation from each language.

After the testing scenario was executed, the subjects were presented with the SUS questionnaire and rated the application. The result obtained by the test application was 86.3, which is *Excellent* according to this rating system. These results, presented in Table 1, showed that an application containing these features is easy to use and understand, gives people confidence while using it and does not require a vast background in using technology. In the first column of the table, we have the questions and on the right-hand side we have the qualifiers from 1(strongly disagree) to 5(strongly agree) together with the percentages of responses given by the subjects for each question and qualifier.

Table 1: SUS Questionnaire responses with score percent

Question: Qualifiers	**1**	**2**	**3**	**4**	**5**
I think that I would like to use this system frequently.	0%	23.1%	1%	30.8%	38.5%
I found the system unnecessarily complex.	61.5%	15.4%	15.4%	7.7%	0%
I thought the system was easy to use.	0%	7.7%	0%	30.8%	61.5%
I think that I would need the support of a technical person to be able to use this system.	61.5%	38.5%	0%	0%	0%
I found the various functions in this system were well integrated.	0%	0%	15.4%	23.1%	61.5%
I thought there was too much in- consistency in this system.	53.8%	23.1%	15.4%	7.7%	0%
I would imagine that most people would learn to use this system very quickly.	0%	0%	7.7%	0%	92.3%
I found the system very troublesome to use.	69.2%	23.1%	7.7%	0%	0%
I felt very confident using the system.	0%	0%	7.7%	38.5%	53.8%
I needed to learn a lot of things before I could get going with this system.	69.2%	30.8%	0%	0%	0%

4. Conclusions

This paper presented a generic solution intended to come in the help of the developers, which can be used in already existing applications and enhance them with features designed to make the application easy to use by people with disabilities. The solution incorporates two components: one for the back-end and the other for the front-end side. The back-end component has a microservices-based architecture hinged on Function as a Service paradigm, that can be accessed through an API end point or as a building block for an application, being able to be deployed in the developer's account through the infrastructure as code capabilities. The front-end side contains client-side data processing and can be included in an application as a library.

As for future implementations, the solution may benefit from the addition of other back-end microservices that can cover a larger spectrum of user needs (like more detail in

image description) and also ensure an encrypted communication channel that could manage sensitive data safely. On the front-end, some benefits could come from a more specialized voice recognition that could be achieved with the implementation of neural networks that are specialized on Romanian language recognition. Altogether these improvements could deliver to the final customer even better and help the technology world to become more inclusive in an easy way.

In the next period, we intend to add applications based on augmented reality, where we are working on facilities that allow voice interaction, in order to have more natural interaction (Irimia *et al.*, 2020; Macariu *et al.*, 2020).

Acknowledgements

This work was supported by project REVERT (taRgeted thErapy for adVanced colorEctal canceR paTients), Grant Agreement number: 848098, H2020-SC1-BHC-2018-2020/ H2020-SC1-2019-Two-Stage-RTD.

References

Adorf, J. (2013). Web speech API. KTH Royal Institute of Technology.

Amazon (2020) Overview of Amazon Web Services. *AWS Amazon Web Services, Inc.* [Online] https://d1.awsstatic.com/whitepapers/aws-overview.pdf

Amazon Web Services (2017). Infrastructure as Code. *Amazon Web Services, Inc.* [Online] https://d0.awsstatic.com/whitepapers/DevOps/infrastructure-as-code.pdf

Baboi, M., Iftene, A. and Gîfu, D. (2019). Dynamic Microservices to Create Scalable and Fault Tolerance Architecture. *Procedia Computer Science, 159*, 1035-1044.

Brooke, J. (1986). SUS - A Quick and Dirty Usability Scale. *Usability Evaluation in Industry*, 189-194.

Calancea, C.G., Miluț, C.M., Alboaie, L. and Iftene, A. (2019). iAssistMe - Adaptable Assistant for Persons with Eye Disabilities. *Procedia Computer Science, 159*, 145-154.

Filimon, M., Iftene, A. and Trandabăț, D. (2019). Bob - A General Culture Game with Voice Interaction. *Procedia Computer Science, 159*, 323-332.

Heymann, J., Offermann, U. and Zdrahal, P. (2001). Communication between client and server computers via http, method, computer program product and system. *Patent US7606901B2* https://patents.google.com/patent/US7606901B2/en

Irimia, C.I., Matei, M., Iftene, A., Romanescu, S.C., Lipan, M. R. and Costandache, M. (2020). Discover the Wonderful World of Plants with the Help of Smart Devices. In *Proceedings of the 17th International Conference on Human-Computer Interaction RoCHI 2020*, 22-23 October, 2020.

Kirkpatrick, A., O'Connor, J., Campbell, A. and Cooper, M. (2018). Web Content Accessibility Guidelines (WCAG 2.1). https://www.w3.org/TR/WCAG21/

Macariu, C., Iftene, A. and Gîfu, D. (2020). Learn Chemistry with Augmented Reality. *Procedia Computer Science*, 176, 2133-2142.

Malakhov, K., Kurgaev, O. and Velychko, V. (2018). Modern RESTful API DLs and frameworks for RESTful web services API schema modeling, documenting, visualizing. ArXiv, abs/1811.04659.

Miluț, C., Iftene, A. and Gîfu, D. (2019). Iasi City Explorer - Alexa, what we can do today? In *Proceedings of the 16th International Conference on Human-Computer Interaction RoCHI 2019*, 17-18 October, Politehnica University of Bucharest, Romania, 139-144.

APPROACHES IN ASSESSING THE CREDIBILITY OF ONLINE INFORMATION

MIRCEA PETIC[1,2], ADELA GOREA[2], INGA ȚIȚCHIEV[1]

[1]*Vladimir Andrunachievici Institute of Mathematics and Computer Science,*
[2]*Alecu Russo Balti State University,*
mircea.petic@math.md, adela.gorea@usarb.md, inga.titchiev@math.md

Abstract

In this paper, we discuss the issue of data credibility, an important one that is specific not only to digital media. We present the main actors and projects concerning the assessment of online information credibility. Then we underline the recent research approaches in assessing credibility of online information. As a result, we have pointed out that an important aspect constitutes the availability of comprehensive dataset for experiments assessing the credibility of information. Not all datasets available for research are developed from the same types of source of information. Some are collected from online news sites, others from social media messages. We note that all datasets presented in the paper are for English, without identifying suitable datasets for research for other languages. In the end, we explore the possibilities of the tools for assessing the credibility of information. The majority of tools are focused mainly on information evaluations based on experts from journalism.

Key words — data credibility, datasets, fake news, social networks.

1. Introduction

Nowadays, people use various information sources in order to be aware of what is happening in the world, to make different decisions, to form different opinions, to develop diverse ideas and values. The importance of data management can be huge: improper management of data can have repercussions that society should avoid. Data credibility is one of the main issues when data comes from unreliable sources. Finding credible data is a task of great interest to those seeking information.

The transition from "the age of information" paradigm to "the age of reputation" should be taken under consideration once attempting to avoid fake news and alternative misinformation techniques that clearly aim to manipulate.

Recently, social media have developed as an important means for the high-speed distribution of news content at a low price. It was found that the number of users looking for news on social networks increased from 49% to 62% from 2012 to 2016[1]. A report by the Jumpshot Tech blog shows that news distributed on Facebook accounted for 50% of total traffic to fake news sites and 20% of total traffic to reputable sites (Perez-Rosas *et al.*, 2018). As almost 62% of the adult population in the US is informed on social networks, it is necessary to identify this false content in online sources.

[1] journalism.org/2016/05/26/news-use-across-social-media-platforms-2016 (last accessed 17 Dec 2020)

The aim of the present article is to identify the functional mechanisms for designing a credibility assessment system for the Romanian language. For this purpose, the paper is structured as follows: first, we identify the actors that are involved in different projects concerning the assessment of online information (Section 2), and then some recent research aspects connected with language behaviours, crowd-source methodology and social context are discussed (Section 3). Section 4 is concerned with the description of available datasets for research in assessing the credibility of information. Even though all the datasets presented are for English, the data collection sources were still different. This allows different approaches to be applied to different types of datasets. Section 5 is devoted to the exploration of tools for assessing the credibility of information. However, in addition to the tools used by experts in the field of journalism to evaluate information, several tools use computational intelligence.

2. Online information credibility assessment communities

The fight against online misinformation is not a task that can be accomplished by a single organization, but needs collaboration to create common concepts, ensuring transparency of decisions[2]. That is why this topic has reached the level of debate in the European Commission. The first result is the approval of a Code of Practice on Disinformation[3]. The Code was approved and signed by the online media platforms Facebook, Google, Twitter, Mozilla, as well as by advertising agencies and the advertising industry in October 2018, and the signatories came up with the ways to apply the Code. Microsoft accepted the code in May 2019, while TikTok joined it in June 2020.

Another initiative is The Credibility Coalition[4], which brings together journalists, researchers, academics, students, policy makers, technologists and employed non-specialists and aims to develop common standards for information credibility by incubating activities and initiatives that gather people and organizations from a range of backgrounds.

The Credibility Coalition website contains a database of projects that have taken place with a view to improving the quality of online information[5]. We also identified a project that is carried out in the Republic of Moldova with the title *ProFact Moldova Network*[6]. This program will build a sustainable network of journalists in Moldova who can test and refine different approaches in creative storytelling, credible media distribution and viral outreach in response to local needs.

There are several other relevant projects. Co-Inform[7] is an project that involves researchers from distinct organizations and SMEs in seven European countries. Its objective is to create tools to foster critical thinking and digital literacy for a better-informed society. These tools will be designed and tested with policymakers,

[2] https://ec.europa.eu/digital-single-market/en/tackling-online-disinformation
[3] https://ec.europa.eu/digital-single-market/en/news/code-practice-disinformation
[4] https://credibilitycoalition.org/
[5] https://credibilitycoalition.org/credcatalog/
[6] https://www.icfj.org/our-work/anti-viral-media-squad-moldova-profact-moldova-network
[7] https://coinform.eu/about/the-project/

journalists, and citizens in three different EU countries. One of the tools developed in this project is MisinfoMe[8]. This tool performs an analysis of Twitter profiles and the followed accounts, with the goal of measuring the influence of misinformation on the life of the citizens.

One more significant platform is The International Fact-Checking Network[9], a department at the Poynter Institute devoted to bringing together fact-checkers worldwide. This network approved a Code of Principles, that is a series of commitments organizations abide by promoting excellence in fact-checking. In the list of entities signing the Code of Principles we can identify many online media sources, including StopFals from the Republic of Moldova[10]. The purpose of the StopFals Campaign consists in diminishing the effects and impact of propaganda and manipulative information that distorts reality, disseminated through various means of communication by media institutions and other politically controlled structures, and developing the capacities of citizens to analyse critically the received information[11].

3. Recent research in assessing the credibility of information

For many years, significant efforts have been made to assess the credibility of online information. Finding erroneous online information consisting only of text is a challenge, as this information is often composed in order to misinform readers, without being able to differentiate it from real news that are written in textual format (Chowdhury *et al.*, 2020). That is why it is important in this situation to try to identify the properties of the text, such as writing style and content, which can help to distinguish between real and fake news articles. The basic idea in this regard is that language behaviours, such as the use of punctuation, word choice, part of speech, and the emotional value of a text are practically involuntary. They are therefore beyond the control of the writer, thus revealing important information on the essence of the text (Perez-Rosas *et al.*, 2018).

Therefore, there are several options for assessing and evaluating credibility that have been tested, from publishing initiatives to establish credibility, to technologies for automatically marking fake news and assessing the credibility of content. There are approaches to hire information verifiers or professional experts, and even campaigns[12,13] to improve literacy or crowd-source annotations (Zhang *et al.*, 2018).

Although the crowd-source methodology seems to be effective in assessing the credibility of online information, there are still situations when the crowd does not have enough relevant information, suggesting that a basic expertise in the crowd is needed. Hence, it is important to see what would be the conditions under which crowdsourcing is a solution for assessing the credibility of online information (Bhuiyan *et al.*, 2020).

[8] https://misinfo.me/misinfo/about
[9] https://ifcncodeofprinciples.poynter.org/
[10] https://ifcncodeofprinciples.poynter.org/application/public/stopfalsmd/1DAF5122-1926-CC13-4C9C-70833CB289E3
[11] https://stopfals.md/ro/about-us
[12] The Trust Project: https://thetrustproject.org
[13] Credibility Coalition: http://credibilitycoalition.org

Thus, there are approaches that use not only the content of the news, but also the social context. The social context for online information is often extracted from social media networks (Chowdhury *et al.*, 2020). For the dissemination of news in the online environment, social media networks appeal to users from two points of view: (1) low costs, easy access, rapid dissemination of information and (2) dissemination of poor-quality news (often intentionally misleading). A lot of research has been done on Twitter and yet there is still no automatic way to find out in real time how to monitor user credibility and message credibility in this network. Recent experiments showed that the mixed approach of combining machine learning in computational linguistics with processing approaches in social networks seems very promising (Iftene, 2019).

We will note that Twitter is not a universal social network, nor is it the most popular in all countries of the world. For example, in the Republic of Moldova there is another ranking according to the number of users of social networks. Gramatic Social Media Report for 2018[14] states that the citizens of the Republic of Moldova have the most accounts on Facebook[15] (1.100.000 users) followed by Odnoklasniki[16] (809.148 users), Instagram[17] (610.000 users), Vkontakte[18] (203.300 users) and LinkedIn[19] (174.218 users).

Data to be processed with machine learning need to be convincing; that is why the importance of data sets for experiments is essential. Further on, we will present several data sets that are used in research to assess the credibility of information.

4. Available datasets for research in assessing the credibility of information

Although there are several approaches in assessing the credibility of online data, the lack of representative fake news datasets is a real obstacle in the full research of the discussed topic. Even if there are certain datasets of news, they often lack certain important characteristics for the study, such as: news content, social context and spatial-temporal information.

All these types of content are combined in the FakeNewsNet dataset[20] that was collected by means of FakeNewsTracker tool (Shu et al., 2020). FakeNewsNet is valuable in fathoming a few issues: testing with a few approaches in recognizing fake news, understanding how they spread on social systems; worldly data permit the consider to distinguish wrong news early; it is conceivable to think about the conveyance prepare in social systems, beginning with those who attempt to persuade of the significance of the news (Shu et al., 2020). Data for FakeNewsNet are gathered from two credible sources (BuzzFeed and PolitiFact). The data contain the text of the news and actions on social media, for example the user who posts/shares the news on Twitter. Speaking in statistical terms, FakeNewsNet contains data about 39,122 users, 62,499 engagements,

[14] Gramatic Social Media Report: https://gramatic.md/blog/gramatic-social-media-report-2019/
[15] Facebook: https://www.facebook.com/
[16] Odnoklasniki: https://ok.ru/
[17] Instagram: https://www.instagram.com/
[18] Vkontakte: https://vk.com/
[19] LinkedIn: https://www.linkedin.com/
[20] FakeNewsNet: https://github.com/KaiDMML/FakeNewsNet

1,209,494 social links, 422 candidate news (211 true news and 211 false news) and 100 publishers (Shu *et al.*, 2019).

In addition to FakeNewsNet, there are other datasets for detecting fake news, few of which contain other features than the linguistic ones, although the social aspect would be very important for this purpose. However, some of the ones presented below will include not only news content, but also social content, and spatio-temporal information.

- BuzzFeedNews[21]: This contains 1,627 articles that consider several political points of view. The data is based on information collected from Facebook during the 2016 US elections. The veracity of each article was verified by 5 journalists from BuzzFeed (Shu et al., 2020).

- LIAR[22]: This dataset consists of 12,800 verified texts made in public speeches and social media collected from PolitiFact, and the statements are labelled in six categories ranging from completely false to completely true. (Shu *et al.*, 2020).

- BS Detector[23]: This dataset contains 874 Links to articles that were accumulated from a browser extension for verifying news credibility. BSDetector check all references on a susceptible web page to be incredible and for this reason it is compared with the information on other sites manualy classified in list of domains (Shu *et al.*, 2020).

- CREDBANK[24]: This dataset consists of 60 million tweets. All tweets were accumulated by 30 online annotators in 3 months in 2015. For this reason, a crowdsource tool, namely Amazon Mechanical Turk, was used. All data refers to 1049 real-world events (Shu *et al.*, 2020).

- BuzzFace[25]: This dataset represents the extension of the BuzzFeed dataset. The additional part for BuzzFace consists of comments posted to news articles on Facebook. The dataset accumulated 2263 news articles and 1.6 million comments on this news (Shu *et al*, 2020).

- FacebookHoax[26]: This dataset consists of 15,500 Facebook posts that refers to scientific news (18 pages) and conspiracy pages (14 pages) (Shu *et al.*, 2020).

- NELA-GT-2018[27] is a dataset of more than 713,000 news articles collected during 10 months in 2018. News articles are accumulated from 194 from websites selected by 8 credible sources. Every news article is labelled with one of the 4 characteristics (credibility, transparency, political polarization, and authenticity) (Zhou *et al*., 2020).

- FEVER[28] is a dataset with more than 185,000 claims that are generated and annotated. The process of dataset completion consists of 2 steps: extraction of a

[21] BuzzFeedNews: https://github.com/BuzzFeedNews/2016-10-facebook-fact-check/tree/master/data
[22] LIAR dataset: https://github.com/thiagorainmaker77/liar_dataset
[23] BS Detector: https://github.com/thiagovas/bs-detector-dataset/blob/master/data_final.csv
[24] CREDBANK: http://compsocial.github.io/CREDBANK-data/
[25] BuzzFace: https://github.com/gsantia/BuzzFace
[26] FacebookHoax: https://github.com/gabll/some-like-it-hoax
[27] NELA-GT-2018: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/ULHLCB
[28] FEVER dataset: https://github.com/awslabs/fever

sentence from Wikipedia, and the manual generation claims based on the extracted sentence. The annotation reffers to labelling the claim in 3 categories "supported", "refuted", or "not enough information" (Zhou *et al.*, 2020).

Another dataset for experiments of assessing information credibility was collected at UAIC[29]. It consists of 2,270 tweets (1,248 not credible and 1,022 credible) and some descriptive information (Iftene, 2019).

Moreover, there is a dataset that facilitates reliability assessment of news on COVID-19. This dataset consists of 2,029 news articles on coronavirus and 140,820 tweets that illustrate the way of spread of these news articles on Web (Zhou *et al.*, 2020).

Table 1: Statistics on available datasets for information credibility assessment

Dataset Name	**Dimension**	**Linguistic info**	**Social-context info**	**Spatial info**
FakeNewsNet	422 articles	+	+	+
Buzz Feed News	1,627 articles	+	-	+
LIAR	12,800 short articles	+	-	-
BS Detector	874 Links to articles	+	-	-
CREDBANK	60 million tweets	+	+	+
BuzzFace	2263 articles	+	+	-
Facebook Hoax	15,500 posts	+	+	-
NELA-GT-2018	713,000 articles	+	-	-
FEVER	185,445 claims from Wikipedia	+	-	-
UAIC FII	2,270 tweets	+	+	+
COVID19	2,029 articles 140,820 tweets	+	+	+

Generally speaking, all the datasets can be classified into three categories:

1. datasets with only news content (full articles or short approvals);
2. datasets which contain social media information that refers to user post;
3. datasets which contain both content and social media information.

What we can notice from Table 1 is that the types of sources for the dataset and the type of information available are varied. Some datasets represent only a collection of texts (LIAR, BS Detector), and others contain also information regarding posts or distributions on social media networks. Social media gives us the opportunity to know who liked a news story and who shared it. This is an indicator that can be taken into account when assessing the credibility of information.

Even if the interest for information written in English is high, it would still be interesting and useful to have datasets for such experiments for other languages, for example Romanian. That is why we consider that a possible research of the process of

[29] "Alexandru Ioan Cuza" University of Iași

assessing the credibility of information written in languages other than English is an area of perspective for further research.

Below are some of the tools available to assess the credibility of online information. They were all designed for the English language.

5. Tools for assessing the credibility of information

Several existing tools that assess the credibility of online texts and/or articles are analysed; they focus mainly on user-generated evaluations as experts in journalism.

FactCheck.org is such an information verification platform that was launched in December 2003. On this site, users can ask questions that are usually based on a rumour in politicians' statements. The site team conducts an investigation and provides a detailed explanation. The explanation includes information about who the author of the statement is, when it was released and how the team verified it. The site also has a special function for verifying scientific information.

Politifact.com is another information verification platform and one of the first fact-checking newsrooms in the US, founded in 2007. The group of reporters within this platform monitors the statements and speeches of politicians and denies false information (Stelino, 2019).

Snopes.com is also an information verification platform developed in 1994 and it aims to validate statements, articles, posts, photos on social media. This platform is not limited to simple statements (*e.g.* "true" or "false"), but uses more detailed categories ("true", "false", "partly true, partly false", "largely true", "mostly false", "outdated information", "misunderstood information", etc.).

Fake Bananas is a tool developed by a group of Sworthmore College students. The tool is based on machine learning algorithms and defines credibility with 82% accuracy. The program searches for authorized online publications of articles on the subject of the message and analyzes whether the authors of the articles agree with the formulated idea made in the statement. If trusted sources agree with it, the program evaluates the statement as true. Although the service is not hosted publicly, the code can be used in other projects (Stelino, 2019).

Hoaxy is a tool developed in 2016 by a group of researchers at the Centre for Complex Networks and Systems Research and the Indiana University Network Science Institute. The tool was developed to study how information is disseminated on social media. Focused on checking the fake news, the site generates interactive, colour graphics, so that users can see how messages are spread on Twitter.

NewsGuard is an instrument for source-level evaluation, manually and methodically verifying English news articles, especially published in the US. NewsGuard is a Chrome extension. It works in the browser or appear in some web searches. This tool verifies the information on sites based on the credibility and the transparency of data (Wineburg *et al.*, 2018). Being evaluated by this tool the websites receive an overall score (the sum of points for each criterion) and a tag. The tag corresponds to a reliable, negative, satirical or platform (blogs, user-generated content or social networks) category. The total score

of credibility and transparency is a maximum of 100. The authors of NewsGuard consider that the site is safe when it accumulates at least 60 points.

My Web of Trust (WOT) is a reputable crowd-sourced service that provides evaluations of websites through a browser extension. It offers two components – trust and security – in terms of a score and a measure of trust. Users can view general ratings and comments from the community and provide their own ratings. The user can also rate it and leave a review based on his personal impressions. WOT works in two ways: real-time protection and manual protection. Real-time protection informs you about online threats (Prochazka and Schweiger, 2018). If you find a site that does not have a reputation rating yet, you can ask the WOT community to rate that site. For ratings and reviews, WOT uses smart algorithms and manual verification to detect and remove fake reviews and can also be used to check blogs and social networks (Twitter, Facebook, Google+).

Although a significant number of scientific papers address many aspects of the topic under discussion, few of them have researched methods of measuring data credibility. There is almost no research that would propose algorithms for assessing the credibility of content that would allow the automation of solving this problem to an extent acceptable (accurate and useful) by users.

We note that although some articles presented complex approaches to solving the problem of online information credibility (using texts, social networking relationships and crowd-source approaches), the available tools promoted online seem to be targeted tests on certain solutions for concrete purposes and not general.

6. Conclusions

The subject of assessing the credibility of online data is of great importance as the European Commission and large companies representing online platforms are involved in dealing with this provocation. It is nice to be aware that the Republic of Moldova is involved in projects and sources of online media in the fight for credible information in the online space.

However, we must recognize that currently only journalists are involved in checking credibility of online information in the Romanian language and there are no information technologies that could automatically assess the credibility of online sources. Unfortunately, no sufficiently complex datasets have been identified that would allow experiments to be performed for the Romanian language with reference to the evaluation of the credibility of online sources in real time.

Likewise, we do not yet have tools that would automatically process online sources for the Romanian language and verify the level of credibility of their information.

However, we consider that the mixed approach, which consists of combining machine learning in computational linguistics with processing approaches in social networks, seems to be very promising.

Acknowledgements

This article was written as part of the research project 20.80009.5007.22 "Intelligent information systems for solving ill-structured problems, processing knowledge and big data".

References

Bhuiyan, M.M., Zhang, A., Sehat, C. M. and Mitra, T. (2020). Investigating 'Who' in the Crowdsourcing of News Credibility. *Proceedings of Computation+Journalism Symposium (C+J'20).* ACM, New York, NY, USA.

Chowdhury, R., Srinivasan S. and Getoor, L.C. (2020). Joint Estimation of User And Publisher Credibility for Fake News Detection. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, October 2020, 1993–1996.

Iftene, A. (2019). Exploiting Social Networks. Technological Trends. Habilitation Thesis submitted at "Alexandru Ioan Cuza" University, December 2019.

Perez-Rosas, V., Kleinberg, B., Lefevre, A. and Mihalcea, R. (2018). Automatic Detection of Fake News. In *Proceedings of the International Conference on Computational Linguistics*, COLING 2018, New Mexico, NM, August 2018, 3391–3401.

Prochazka, F. and Schweiger, W. (2018). How to measure generalized trust in news media? An adaptation and test of scales. In *Communication Methods and Measures*, 13(1), 2018. 26–42.

Shu, K., Mahudeswaran, D., Wang, S., Lee D. and Liu H. (2020). FakeNewsNet: A Data Repository with News Content, Social Context, and Spatiotemporal Information for Studying Fake News on Social Media. *Big Data*, 8, 3, 171-188.

Shu, K., Wang, S. and Liu, H. (2019). Beyond News Contents: The Role of Social Context for Fake News Detection. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM 2019, 312–320.

Stelino, M. (2019). 8 resources for detecting mis- and disinformation, In *International jurnalists network*, https://ijnet.org/en/story/8-resources-detecting-mis-and-disinformation.

Wineburg, S., Mcgrew, S., Breakstone, J. and Ortega, T. (2018). Evaluating information: The cornerstone of civic online reasoning. In *Stanford Digital Repository*. Retrieved January, 8:2018.

Zhang, A., Ranganathan, A., Metz, S. E., Appling, S., Sehat, C. M., Gilmore, N., Adams, N. B., Vincent, E., Lee, J., Robbins, M., Bice, E, Hawke, S. and Karger, D. (2018). A Structured Response to Misinformation: Defining and Annotating Credibility Indicators in News Articles. In *Companion Proceedings of the The Web Conference*, WWW 2018. Lyon, 603–612.

Zhou, X., Mulay, A., Ferrara, E. and Zafarani, R. (2020). ReCOVery: A Multimodal Repository for COVID-19 News Credibility Research. In *Proceedings of the 29th*

ACM International Conference on Information & Knowledge Management, CIKM, October 2020, 3205–3212.

AUTOMATIC FAKE NEWS IDENTIFICATION SYSTEM

CIPRIAN-GABRIEL CUȘMULIUC, IOAN SAVA, DIANA-ISABELA CRAINIC, LUCIA-GEORGIANA COCA AND ADRIAN IFTENE

Faculty of Computer Science "Alexandru Ioan Cuza", University of Iasi, {cusmuliuc.ciprian.gabriel, ioan.sava, diana.crainic, coca.lucia.georgiana, adiftene}@info.uaic.ro

Abstract

News sharing in social media has become a norm in our society. Every day, we are met with hundreds of new posts, each presenting different statements or facts that sometimes we take for granted, without verifying their veracity. This daunting task of news and claim verification is less feasible if we consider that, day by day, the information quantity available on the web increases drastically, hence the problem at hand of identifying fake news. The present paper aims to analyze different techniques of detecting fake news and measure their performance, but also proposes a web application which pulls tweets and labels them accordingly so the user can avoid misinformation.

Key words — CNN, Fake news, Naïve Bayes, SVM, Twitter.

1. Introduction

Increase in social networks popularity has led users to conduct multiple activities in social media, such as: communication, blogging, vlogging and news reading, the latter being the subject of our discussion. News sharing can be done very easily nowadays and the fact that claim verification is very difficult due to the vast quantity of information can lead to a well-known problem: misinformation. This problem is made worse by the fact that many respectable news outlets also copy stories found on social platforms. Thus, the problem can spiral out of control very easily, leading users to believe fake stories, assuming their veracity was previously checked.

Over the years there have been different approaches to the classification of fake news, from linguistic approaches to social network approaches. Linguistic approaches focus on creating statistics on n-grams (Hadeer *et al.*, 2017) or sentence transformation into parsing trees and analysis of anomalies (Perez-Rosas *et al.*, 2017). In social networking approaches, knowledge networks are exploited to identify the lie (Idehen, 2017) and the fact that users are forced to authenticate when using the social network provides increased confidence to the data that appears here (Shu *et al.*, 2018; Wu and Liu, 2018). In addition to these approaches, there were approaches that aimed to identify the credibility of news and information on the Internet (Atodiresei *et al.*, 2018; Iftene *et al.*, 2017; Iftene *et al.*, 2020).

Considering the aforementioned approaches, in this paper we decided to focus our attention on creating artificial intelligence models in combination with linguistic techniques. The sections are as follows: Section 2 will focus on related work and common applications available for fake news detection, Section 3 describes the proposed solution, Section 4 aims to measure the performance of each component and

of the whole system whilst comparing it to others and, in the end, and we will draw a conclusion and present future work.

2. Related Work

2.1. Papers

In recent years there have been notable efforts in combatting fake news. We analysed a few state-of-the-art systems in order to better develop our model; in this section we will discuss some approaches that have influenced our decisions.

First, Hadeer *et al.* (2017) aim to "investigate and compare two different features extraction techniques and six different machine classification techniques". They compare n-grams with multiple machine learning algorithms; the tested models are: SVM, LSVM, KNN, Decision Tree, SGD and Logistic Regression (n-grams tested range from unigram to four-gram). The best result yielded is "using Term Frequency-Inverted Document Frequency (TF-IDF) as feature extraction technique, and Linear Support Vector Machine (LSVM) as a classifier, with an accuracy of 92%". The dataset consists of 12,600 fake news articles from Kaggle.com and 12,600 truthful articles.

Perez-Rosas *et al.* (2017) take a different approach and "present several exploratory analyses on the identification of linguistic differences in fake and legitimate news content". They start by extracting linguistic features, such as: n-grams, punctuation, psycholinguistic features, readability and syntax. Punctuation is created from a software called Word Count Software (LIWC, Version 1.3.1 2015) (Pennebaker *et al.*, 2015) and "this includes punctuation characters such as periods, commas, question marks and exclamation marks". Psycholinguistic features analysis is done using the same LIWC software and it contains a large lexicon of word categories that represent psycholinguistic processes (*e.g.*, positive emotions, perceptual processes), summary categories (*e.g.*, words per sentence), as well as part-of-speech categories (*e.g.*, articles, verbs). They measure readability using the following features: number of characters, complex words, long words, number of syllables, word types and number of paragraphs. The best F1 score is derived from Readability and Support Vector Machine (SVM); on the second place, it is a combination of all the features and SVM. Using the information above we decided to build a complex feature set from our dataset and use TF-IDF in combination with SVM as well as Convolutional Neural Network (CNN) Long Short-Term Memory (LSTM).

2.2. Common applications

Fake News Detector

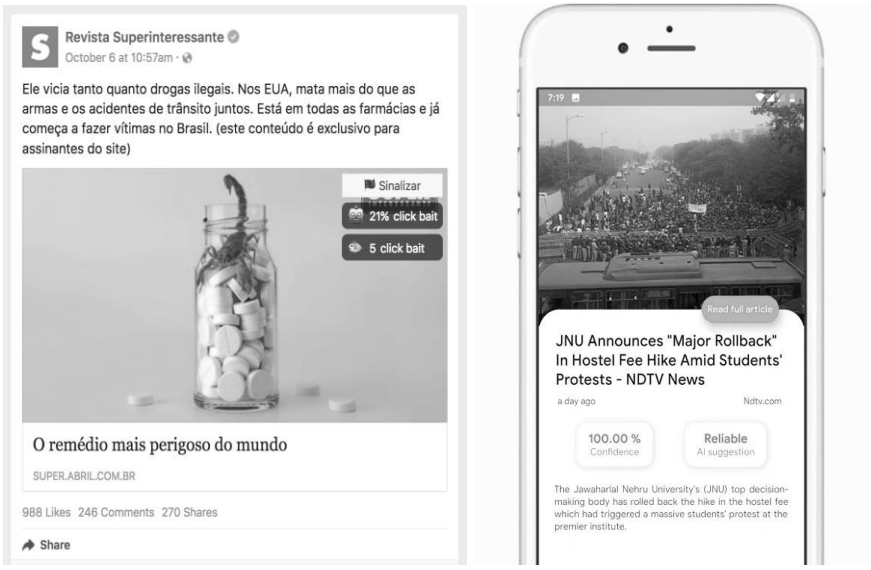
Fake News Detector[1] is a Chrome extension (see Fig. 1.a) which labels posts as: Fake News, Click Baits and Extremely Biased news. It uses a combination of human flagging and automatic learning to quickly flag posts. Compared to our application, it has a much larger database because it was launched in 2018 and user flagging has increased the

[1] https://fakenewsdetector.org/en

dataset. Unfortunately, it lacks an accuracy metrics and relies on a very basic Naïve Bayes algorithm that classifies the Facebook posts making its accuracy questionable.

NewsChase

NewsChase[2] (see Fig. 1.b) is an Android application that measures the imaginative writing styles in a given news article. It basically uses a machine learning algorithm to measure if the article is a work of fiction or not. The authors do not provide details on the implementation nor the methods used or the dimension of the dataset. Its main selling points seem to be the Android application, an easy to use UI that shows recent news from different sites and a score of confidence with an Artificial Intelligence (AI) suggestion tag. The AI is basically made up of 3 possibilities[3]: (1) "Reliable: if the article is written in an informative style"; (2) "Alright: if the article has moderate use of imaginative writing styles"; (3) "Beware: if the article may have manipulative content. Reading with caution is advisable". The author explicitly states on the application website that the algorithm "does not check factual information, and only analyses the writing style".



(a) (b)

Figure 1: Fake News Detector Application (a), NewsChase example (b)

3. Proposed solution

The proposed solution is a system capable of pulling Twitter data, analysing and labelling them as fake or not fake. The system also provides users with a dashboard so that they can easily visualize the information and the label attached to each tweet; if the post has been processed, they will see the according label, and else the system will show a pending status. In this section we will describe the inner workings of the system.

[2] https://newschase.github.io/
[3] According to https://play.google.com/store/apps/details?id=com.iiserb.fictonews&hl=en

3.1. Crawling

In order to show the tweets to the user, we first needed to develop a mechanism that would query for them and insert them in a database. This process consists of using the Twitter API to extract and insert tweets in a cloud Mongo[4] database, from where they can be called upon for filtering and further processing by the other modules. To access tweets, the Twitter API is used via a Python package called Tweepy[5]. The tweets are stored in JSON format with the following fields: *text, author, dat*e and *hashtags*. In order to maintain data consistency, before inserting them in the database, we checked for tweet duplication and verified if the tweet is a piece of news; the news classification of the tweet was done using the algorithm described in Subsection 3.2.

3.2. Filtering

As mentioned in the previous subsection, before saving a tweet, there was a need to identify the category to which it belongs, so we know which tweets continue to the classification process (only news is classified). We decided to create four big tweet categories that we believe can summarize very well the activity on the social media platform; they are: *news*, *jobs*, *ads* and *others*. In order to label tweets with various categories, we proposed two methods which are a **Dictionary-Based Filtering Algorithm** and a **Naïve Bayes classifier**.

Dictionary-Based Filtering Algorithm: This method uses a series of predefined word lists (manually collected), which appear mainly in the text or among the tags of the tweets in the considered categories. When the algorithm receives a tweet as input, it takes its text and tags and converts them into a sequence of tokens. Then, for each token in the sequence, it checks if the token is among the predefined lists words. Depending on the first token that appears in a predefined list, the tweet will be properly categorized. If no token appears in any list, the tweet is considered *others*. Some examples of words associated to each category are the following:

- for the category *ads*, the tweet might contain “ad”, “sell”, “promotion” and so on.
- for the category *jobs*, the tweet might contain “apply now”, “hiring”, “career” and so on.
- for the category *news*, the tweet might contain “breaking news”, “news”, “trending news” and so on.

Since this approach proved to be unreliable, as we will see in the results section, we decided to try a machine learning approach, using a Naïve Bayes classifier.

Naïve Bayes Algorithm: We developed a dataset of 800 tweets that were manually annotated in order to train the classifier and obtain the tweet categories. The dataset was comprised of the following fields: *tweet_id*, *tweet_text*, *tweet_tags* and *label*. An example tweet can be the following: *tweet_id*: “1249451240603426825”, *tweet_text*: “The coronavirus was already spreading within America’s borders by the time the China travel restrictions went into effect in early February”, *tweet_tags*: “coronavirus”, *label*:

[4] https://www.mongodb.com/
[5] http://docs.tweepy.org/en/latest/

"news". For the dataset we decided to use tweets tailored for this year's events, so that the algorithm will be fit for today's type of news, which seem to be much more sensational and urgent. In the dataset, there are tweets related to COVID-19, Trump, Remote Job ads and so on (things that one year ago were not so prevalent). The machine learning model simply reads the data, tokenizes it, trains on it and then it can emit a category for input tweets.

3.3. Pre-processing

Data pre-processing is a vital step in the process of feature extraction. We used multiple techniques such as: cleaning stop words, emoticon deletion, duplicate words deletion and uppercase word identification. We will discuss each of them in this section.

Stop words: We decided to remove stop words as they would not help us in the classification process; moreover, by eliminating them we gain performance since the algorithms' input is reduced. In order not to alter the phrase meaning, we took into consideration that there are cases where removal of stop words would result in an erroneous output. For instance, the sentence: "I didn't like the product" is negative, and when we eliminate the stop words, the new sentence: "like product" is positive, hence the need to skip this sentence. We observed that after fixing the removal of stop words that changed the meaning of the phrase, the overall accuracy increased. In order to aid our work, we used a list of English stop words found on GitHub[6].

Emoticons deletion: Emoticons - short for emotion icons - are undeniably one of the most widespread means of communication. So much that the "*Face with Tears of Joy*"[7] was dubbed Oxford Dictionaries 2015 "Word of the Year"[8]. Emoticons are represented in text blobs as Unicode blocks, usually in multiple characters, making them difficult to handle. As an example, the flag of Scotland uses 7 component characters, "\U0001f3f4 \U000e0067 \U000e0062 \U000e0073 \U000e0063 \U000e0074 \U000e007f", in full escaped notation. We deemed worthwhile to use a Python library called Demoji[9] to handle their extraction, in order to ensure stability and consistency of the classification algorithms. Even though emoticons provide valuable information (the emotions involved in the communication act), they proved difficult to model in the system, hence we decided to remove them.

Duplicate words deletion: Duplicate words in a sentence or in a context are redundant and can cause a decrease in performance by sending redundant data through the system. For example, consider the sentence: "I I I I I I I like dogs and cats". It is easy to see that the word "I" is redundant; instead, we keep only one copy: "I like dogs and cats". Duplicate words do not have to be the same, they can be synonyms or extremely related. For example, consider the sentences: "I like adore love dogs and cats". *Which one of these words do we erase?* They have slightly different meanings. *How similar do two words must be in order to erase one of them?* The answer is not very straightforward, nor is the solution. In order to calculate the similarity between two words, we imply a

[6] https://github.com/Alir3z4/stop-words
[7] https://emojipedia.org/face-with-tears-of-joy/
[8] https://time.com/4114886/oxford-word-of-the-year-2015- emoji/
[9] https://pypi.org/project/demoji/

solution based on NLTK[10]. If the words are identic, then the computed similarity score is 1; an example is the following: (cat, cat) →1.

If the words are not the same, using NLTK's Synset API, we will create a synonym ring for each of the words, meaning we will get a list of terms that express the same concept. The idea is to search every word in the synonym ring and if we find a common word, we calculate and return the Wu & Palmer Similarity (Wu and Palmer, 1994). Experimentally, we concluded that the similarity factor should be high, around 0.9, meaning we must be very careful with word removal.

Uppercase words: Uppercase words can yield important information in the detection of fake tweets. If we consider the frequency of uppercase words and lowercase ones, we can extract a ratio that can provide meaningful insight into the trustworthiness of the news, meaning we can approximate if the author is aiming for sensationalism or for genuine content sharing. Exaggerated and provocative headlines with excessive use of capital letters or emotional language are serious red flags[11]. Yang *et. al.* (2018) finds that the 'Top Fake' words are capital characters, whilst the 'Top Real' words contain many names and motion verbs, expressing the 'who' and the 'what' in the story (which are two important factors in the five elements of news: *when*, *where*, *what*, *why* and *who*). As such, four functions were implemented in order to extract features related to uppercase/lowercase words/letters: *noOfUppercaseLetters* - returns the number of upper-case letters in the given string, *noOfLowercaseLetters* - returns the number of lower-case letters in the given string, *noOfAllCapsWords* - returns the number of all caps words in the given string and *noOfWordsStartingWithCapitalLetter* - returns the number of words starting with capital letters in the given string.

3.4. Dataset

The dataset consists of 10,000 tweets that were manually annotated with 'fake' or 'not fake' labels. They have been split in 2 datasets, one for training that had 7,000 tweets and one for testing, with 3,000 tweets. The main tweet features were: *Date, Tweet_Text, Tweet_Id, User_Id, User_Name, User_Screen_ Name, Retweets, Favourites* and *Class*. An example of such a tweet is the following: *Date*: "2011-08-06 19:59:59", which is in standard date format, *Tweet_Text*: "RT @joeashtonsing New Guidelines for Product Branding at London 2012 Olympics - Bike Europe http://bit.ly/ pFRMqG", which is the tweet content, *Tweet_Id*: "98900000000000000", *User_Id*: "67898811", *User_Name*: "The London bike bot", *User_Screen_Name*: "67898811", *Retweets*: *"2"*, *Favourites*: "3", *Class*: "1", which represent the fact that the tweet was manual classified, and it represents a fake tweet.

In order not to make the algorithm biased toward users with high favourites and retweets count (*i.e.*: only the very popular on Twitter say the truth), the dataset does not contain outliers; the highest retweet count is around 200 and the favourites are around 100, very natural numbers for average Twitter users (from various sources, the average being 278 retweets). Also, users that use many hashtags and reference other users in

[10] https://www.nltk.org/
[11] https://abqlibrary.org/fakenews/factcheck

their tweets are more likely to tell the truth, as what they say is based either on the hashtag (that might be, for example, a trend) or in response to another user.

3.5. Algorithms

In this section, we will discuss the two algorithms used for fake news identification, the first one being CNN LSTM and the second SVM. The motivation behind the choices lies in a similar work (Cusmuliuc *et al.*, 2018), where we used primarily machine learning approaches and, wanting to try a different perspective, we thought of using CNN LSTM so that we might obtain better results. At the same time, we reused SVM as it was one of the best performers and we wanted to try and see how it would work when we changed the feature set.

CNN LSTM Approach

The first algorithm used was CNN with LSTM. In this subsection we describe how the algorithm works and the pipeline for the classification. The first step of the pipeline involved calling the pre-processing module that involved removing stop words, emoticons, duplicate words, numeric characters, hashtags, hyperlinks, punctuation, single characters and usernames (as described in subsections prior). After the pre-processing step, we create an embedding matrix using Word2vec[12] in order to feed it to the neural network. In addition to the tweet sent as a vector, we also provide the algorithm with the sentiment of the tweet, number of uppercase letters, number of words starting with capitals and word rate. The architecture of the model can be seen in Fig. 2.

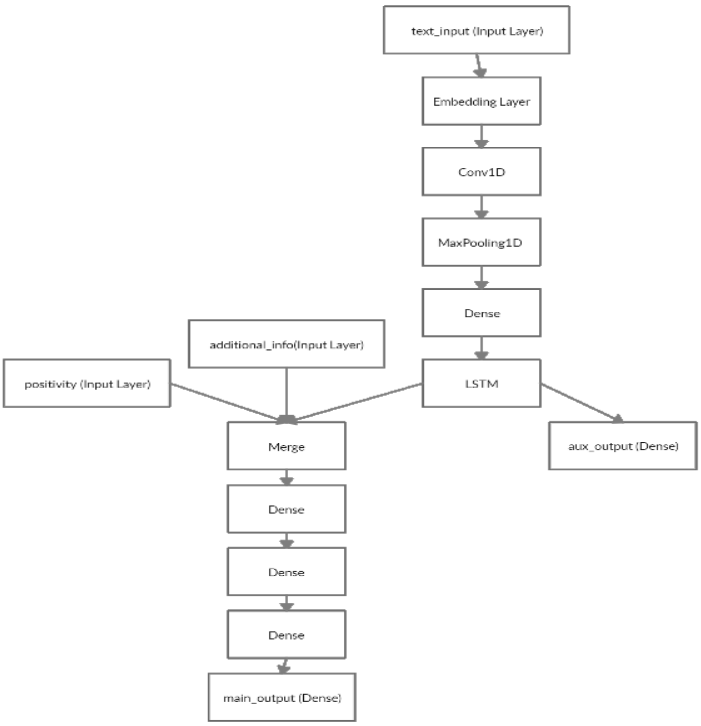


Figure 2: Model architecture

[12] https://pypi.org/project/word2vec-keras/

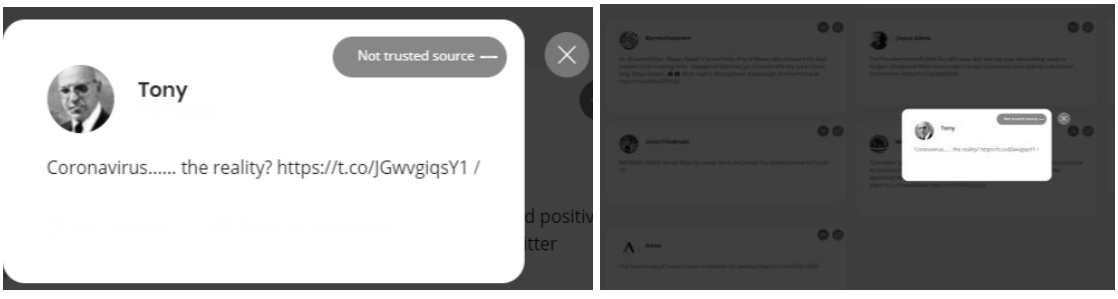
The model starts with an embedding layer, comprised of the calculated embedding matrix resulted from Word2vec; next it goes through a 1-dimension convolution which basically reads from the embedding matrix and the next max pooling layer will down sample, so we can send it to a regular densely-connected neural network layer with a softmax activation layer. After the CNN convolutions, we have the LSTM network that gets input from the latter algorithm; this allows the network to build up internal state and update the weights with back-propagation through time, across a sequence of the internal vector representations of input tweets. After the LSTM, we have 3 layers of densely connected neurons, each layer with 64 neurons that get the input from LSTM and merge them with additional information (sentiment, number of uppercase letters etc.), so that in the end we get the result using a sigmoid activation function.

SVM Approach

The second algorithm used was SVM, as we believed the hyperplane can easily split the tweets in the two classes (*fake* and *real*). We used the same techniques of pre-processing as in CNN LSTM, remove stop words, emoticons, duplicate words, numeric characters, hashtags, hyperlinks, punctuation, single characters and usernames. For feature extraction, we decided on TF-IDF, in order to be able to feed the algorithm a matrix of features. The hyper-parameters for SVM were chosen using Grid Search; we left the algorithm to train and evaluate and arrived at the following best combination: *c* = 0.1, *gamma* = 0.2 and *kernel* = 'rbf'.

3.6. Tweets UI

In order to better aid users with the detection of fake tweets, we decided it is best to also include a UI. We designed a website where users can better see the newest tweets in their network, each tweet having a label attached to it. We took advantage of our cloud architecture, meaning that we were able to push classified tweets from the back-end microservice architecture (Jamshidi *et al.*, 2018) to the front-end, through web sockets[13]. In Fig. 3 (a), we provided an example of a tweet: the user can see the username of the poster, the message and the score given by our system; in this case, the system thinks it is fake. Fig. 3 (b) provides an overview of the whole dashboard where the user can see his most recent tweets.



(a) **(b)**

Figure 3: Example of tweet (a), Example of dashboard (b)

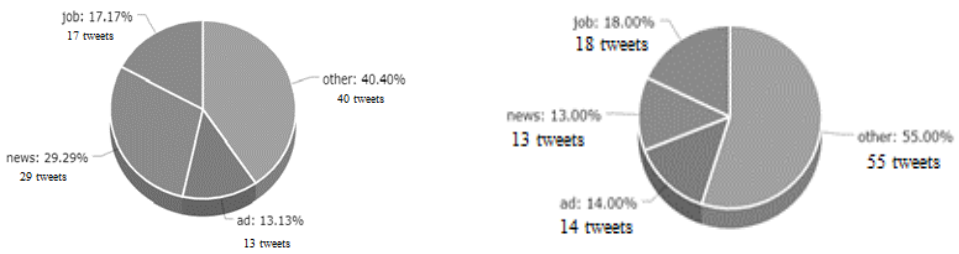
[13] https://tools.ietf.org/html/rfc6455

4. Results

4.1. Experiments on filtering

When testing the proposed algorithms, we made use of 2 datasets of 100 tweets each; the first comprised already-classified tweets from all acknowledged categories (*news*, *job*, *ad* and *others*) and the second tweets labelled as news.

For the first set, we analysed each tweet and then manually assigned a second label, in addition to the existing one. Then we determined the number of tweets in which the first label matched the second one. That number represents the efficiency percentage. For the second set, we again analysed each tweet, and, for each news tweet, we manually assigned a verdict (*yes* or *no*) specifying whether that tweet was indeed news or not. Then we determined the number of tweets in which the verdict was affirmative. That number represents the efficiency percentage. The tweets distribution on categories is presented in Fig. 4 a.



(a) (b)

Figure 4: Tweets distribution on categories (a), Results of Dictionary-Based Filtering Algorithm (b)

Dictionary-Based Filtering Algorithm: On a set of 100 tweets from all acknowledged categories, we noticed a correctness percentage of 62% (out of 100 tweets, 62 were classified with the right label). Another interesting result was noticed when using the algorithm to get 100 tweets related to news. In that situation, out of 100 tweets, 51 were correctly labelled as news. Since the algorithm labels tweets based on several dictionaries of keywords, we can say that the classification manner is a naive one. Moreover, the classification is strongly dependent on the order in which each dictionary is used. For example, if a tweet contains keywords from 2 dictionaries, in our case A and B, depending on the order in which those dictionaries are verified, the tweet can be labelled as being from both A and B. The results can be seen in Fig. 4 (b).

Naïve Bayes Algorithm: On a population of 100 tweets from all acknowledged categories, we noticed a correctness percentage of 64% (out of 100 tweets, 64 were classified with the right label) (see Fig. 5). Another interesting result was noticed when using the algorithm to get 100 tweets related to news. In that situation, out of 100 tweets, 63 were correctly labelled as news.

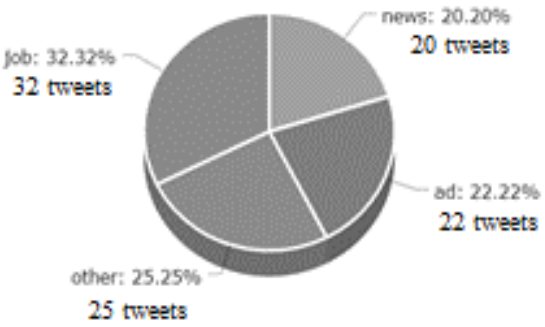


Figure 5: The results of Naïve Bayes Algorithm

In order to work, the algorithm makes use of a sample training data pool. Because of that, the precision of the classification is dependent on the size and variety of the sample pool. The larger the sample pool, the better the results, but if the sample pool lacks variety, then classification errors are still bound to occur. For example, the news label can be assigned to tweets commenting or expressing opinions on different news. Taking all the above into account, we can say that a Naïve Bayes proved to be a favourable approach. Unlike a keyword-based filtration algorithm, which makes use of one or many dictionaries in order to classify the tweets into different categories (because of that, it can often fail), a Naïve Bayes algorithm works with training data, meaning that it labels the received tweets with respect to a population of already labelled tweets. Therefore, even if a misleading tweet might have corresponding keywords or tags, it can be correctly classified based on the algorithm's training data.

4.2. System

In order to test the system, we decided to create a test dataset comprised of tweets pulled from the Twitter API. As such, they are very recent, and we had to manually label them before sending to the system. We collected 100 tweets that are related to very recent news events (Covid, US elections) but also ads, normal chatter and so on. We tried making the test dataset as diverse as possible, so the system can be tested in all scenarios. Some example tweets are the following: „Danny Dyer has clearly never heard the saying 'Better to remain silent and be thought a fool than to speak and to remove all doubt.'. https://t.co/4qijvzBLXt", „The schedule is set for the 2021 JSU Soccer Spring slate...Mark your calendars now! https://t.co/ly1hFRlakt", „Biden extends his lead from 10 to 14 points. @JLPartnersPolls for @Independent https://t.co/rCKzbvw09s https://t.co/UOSjifO9Bn" – it can be seen that the tweets are very diverse, from political news to normal chatter and sports announcements.

We fed the list of tweets to the system, which first pre-processed them removing stop words, emoticons, duplicate words, numeric characters, hashtags, hyperlinks, punctuation, single characters and usernames (as described in subsections prior) and then sent the resulting information to the trained classification algorithms. We managed to get good results: CNN LSTM managed an average of 89% accuracy whilst SVM scored 88%.

5. Conclusions

While trying to find a solution to a complex problem, we found that CNN LSTM has very good accuracy, outperforming SVM by a point. We believe that, with some further fine-tuning, the algorithms can perform even better (especially CNN LSTM, that shows strong potential).

We also designed the system to be user friendly by having a UI, where people can browse tweets whilst being informed about their veracity. For future work, we plan to improve the accuracy of these algorithms by creating a larger training dataset, including a function in the UI where users can report fake tweets that were wrongly classified and to include a module that does inference over a tweet. Also, a potential plan is to include an attention mechanism, such as BERT (Devlin *et al*., 2018).

Acknowledgements

This work was supported by project REVERT (taRgeted thErapy for adVanced colorEctal canceR paTients), Grant Agreement number: 848098, H2020-SC1-BHC-2018-2020/ H2020-SC1-2019-Two-Stage-RTD.

References

Atodiresei, C.S., Tănăselea, A. and Iftene, A. (2018). Identifying Fake News and Fake Users on Twitter. In *Proceedings of International Conference on Knowledge Based and Intelligent Information and Engineering Systems*, KES2018, 3-5 September 2018, Belgrade, Serbia. Procedia Computer Science, 126, 451-461.

Cusmuliuc, C.G., Coca, L.G. and Iftene, A. (2018). Identifying Fake News on Twitter using Naive Bayes, SVM and Random Forest Distributed Algorithms. In *Proceedings of The 13th Edition of the International Conference on Linguistic Resources and Tools for Processing Romanian Language* (ConsILR-2018), 177-188.

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv*, abs/1810.04805

Hadeer, A., Issa, T. and Sherif, S. (2017). Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments ISDDC 2017*, 127-138.

Idehen, K.U. (2017). Exploitation of a Semantic Web of Linked Data, for Publishers. *Open Link Virtuoso Universal Server*.

Iftene, A., Dudu, M.Ș. and Miron, A.R. (2017). Scalable system for opinion mining on Twitter data. Dynamic visualization for data related to refugees' crisis and to terrorist attacks. In *26th International Conference on Information Systems Development*, Larnaca, Cyprus, September 6-8, 2017.

Iftene, A., Gîfu, D., Miron, A.R. and Dudu, M.Ș. (2020). A Real-Time System for Credibility on Twitter. In *Proceedings of the 12th Language Resources and Evaluation Conference* (LREC 2020), Marseille, France, May 2020, 6168-6175.

Jamshidi, P., Pahl, C., Mendonça, N.C., Lewis, J. and Tilkov, S. (2018). Microservices: The Journey So Far and Challenges Ahead. *IEEE Software*, 35 (3), 24–35.

Pennebaker, J.W., Boyd, R.L., Jordan, K. and Blackburn, K. (2015). The development and psychometric properties of liwc2015. Technical report.

Perez-Rosas, V., Kleinberg, B., Lefevre, A. and Mihalcea, R. (2017). Automatic detection of fake news. *Proceedings of the Association for Information Science and Technology*, 52(1), 1-4.

Shu, K., Sliva, A., Wang, S., Tang, J. and Liu, H. (2018). Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter*, 19(1), 22-36.

Wu, L. and Liu, H. (2018). Tracing Fake-News Footprints: Characterizing Social Media Messages by How They Propagate. In *Proceedings of the eleventh ACM international conference on Web Search and Data Mining*, 637-645.

Wu, Z. and Palmer, M. (1994). Verb semantics and lexical selection. In *Proceedings of the 32nd Annual meeting of the Associations for Computational Linguistics*, 133-138.

Yang, Y., Zheng, L., Zhang, J., Cui, Q., Li, Z. and Yu, P.S. (2018). TI-CNN: Convolutional Neural Networks for Fake News Detection, *arxiv, https://arxiv.org/abs/1806.00749.*

WHAT INDICATORS TELL US ABOUT MAKING ACCURATE RANK OF THE BEST PAPER PREDICTIONS

DAN ALEXANDRU[1], ADRIAN IFTENE[1] AND DANIELA GÎFU[1,2]

[1]*A. I. Cuza University, Faculty of Computer Science, 16 Berthelot St., Iași, Romania*
[2]*Institute of Computer Science, Romanian Academy, 2 Codrescu St, Iasi, Romania*

{dan.alexandru, adiftene, daniela.gifu}@info.uaic.ro

Abstract

We propose a pilot research methodology intended to predict the best paper ranking, including machine learning algorithms based on arXiv collection. Our approach plans to link the author's *h-index*, identified in the Semantic Scholar, with the quality of a paper. It is well known that the *h-index* is a significant indicator of research impact realised by one author or a team, based on citation measurement. Out of these considerations of paper ranking use, we will concentrate in this survey only on the one or more of the authors on the search results page, by checking the *h-index* for each of them.

Key words — arXiv collection, *h-index*, machine learning, paper ranking.

1. Introduction

Scientists generally have an emerging formal discursive behaviour (Gifu and Teodorescu, 2014), being largely determined by the formation of academia norms within the working groups. Nowadays, any scientific topic is covered in hundreds of articles. A researcher or a scientist team makes a huge effort to select the best papers about it in order to make progress about a specific topic. To have a realistic point of view based on the quality of the scientific publication means to establish several criteria for determining what is or not important (Cioffi *et al*., 2020; Yokuş and Akdağ, 2019). In this regard, there is a need for balance between comprehensiveness and rigid rules, concrete enough for a real decision (Wickson and Carew, 2014), but flexible enough to be adapted into specificities of different contexts (Szklo, 2006). As we know, published research is growing at an astonishing rate, yet the tools to decide the scientific level of researchers are still unconvincing. At most, we expect excerpts, references/quotes most of the time, sometimes recommendations and code submissions, but rarely Keyphrase Extraction Algorithm (KEA), hierarchical modelling of the topic or meta-analysis researching a particular topic (apart from work that does just that).

The legitimate question of this survey is: *How to decide the research quality published on a specific platform?*

Our approach aims to reduce this gap for a good decision, using part of arXiv collection (scientific papers, LaTeX sources if available) and metadata and improving the features usually extracted from papers. In fact, this approach describes one of the criteria to predict the best paper ranking, by linking the author's *h-index*, identified in the Semantic Scholar, with the quality of a paper.

The paper is structured as follow: Section 2 presents a short background of a number of studies to ranking authors, journals or conferences in order to have a realistic starting point about this topic, while Section 3 describes the data set and the proposed method, which are under implementation, including the architecture system. Section 4 briefly discusses the phases of our method, including five algorithms we encountered in the literature focused on this topic, before drawing some conclusions in the last section.

2. Background

Smart systems require innovative solutions to predict the rank of the best scientific papers. In this context, artificial intelligence (AI)-driven technologies, leveraged by advanced embedded systems, big data, cognitive models, etc. are ready to generate new scientific publishing paradigms (Gupta, 2017). Until now a number of AI-based techniques succeeded to classify the best, innovative and cutting-edge science works using the Web of Science and SCOPUS database (Cioffi, 2020). If we look at the data offered by the arXiv collection, more and more research is published on a monthly basis. Klein *et al.*'s (2019) study shows that the content of the text of scientific papers has generally changed very little from pre-printed versions to published final versions, as in this platform. Either on broader topics, such as Machine Learning (ML) or on more specific subjects, it becomes increasingly hard for researchers to keep up with the literature in their specific field.

An added problem is that popular conferences such as ICML[1], ICLR[2] and NeurIPS[3] are flooded with submissions. This, combined with a limited number of reviewers available, leads to a between the hammer and the anvil kind of situation: (1) increase the number of reviewers (which may decrease the average reviewer expertise, with cases of recent PhD graduates as reviewers) or (2) increase the reviewing quota for each reviewer (hard to give balanced reviews with tens or hundreds of papers to review (Gschwend, 2005)). The second solution becomes even a greater problem when it gets to the rebuttal phase, with responses from the authors that need to be addressed. Current approaches to ranking authors, journals or conferences are based on variations of *h-index*. Yet, when closely examined, *h-index* is unreliable, as shown by Waltman and Eck (2012), Bensman *et al.* (2014) or Horzyk (2014). Meta-analysis has been proven a successful approach when it comes to scientific literature (Lee *et al.*, 2018) and medical research (Lee, 2018), yet it seems a rarely-approached issue in Computer Science or ML research (with a probably sole exception of Henderson and Brunskill (2018)).

3. Data and Method

3.1. arXiv Collection

It is well known that the *h-index* is a significant indicator of the research impact realised by one author or a team, based on citation measurement, along with the *c-index* and *c'-index* indexes (Ciaccio *et al.*, 2019). Out of these considerations of paper ranking use,

[1] International Conference on Machine Learning (ICML), https://icml.cc/
[2] International Conference on Learning Representations (ICLR), https://iclr.cc/
[3] Conference on Neural Information Processing Systems (NeurIPS), https:/nlps.cc/

we will concentrate in this pilot study only on the one or more of the authors on the search results page, by checking the *h-index* for each of them. arXiv is one of the most popular platforms when it comes to pre-publishing articles and hosting open-access articles (Klein *et al*., 2019). It has collected a good part of the published research from 1993 to present and has a representative part of Computer Science research under domains such as AI (cs.AI), Computational Linguistics (cs.CL) and Machine Learning (cs.LG). arXiv has offered multiple options to access their data:

- arXiv API – made primarily for atomic queries, by discarding results of previous queries if one of the subsequent queries fail and cause the rollback;
- arXiv OAI (Open Archives Initiative)-PMH interface – made for bulk metadata access. Metadata for arXiv articles may be reused in non-commercial and commercial systems;
- arXiv S3 buckets – host the full text (s3:///arxiv/pdf/) and sources data dump (s3:///arxiv/src/). In fact, arXiv uses Amazon S3 with the "requester pays" option. The storage containers of S3 are called "buckets" and they are addressed in an URI style: s3://arxiv/pdf/arXiv_pdf_ manifest.xml;
- arXiv Export Mirror – allows scrapers/bots to use this domain in order not to impact the users on the main arXiv site.

We perform a sync with arXiv's bucket (copy if archive not present) on our raw bucket and save the sync logs. We harvest the OAI-PMH metadata and save it in the raw bucket.

3.2. Data Cleaning Process

After a sync is made with arXiv's bucket, we unzip the data in a spot instance and start to process it. We check whether the paper-wise archives are either source archives or PDF archive. If no source is present in the S3 sources folder, the PDF will be offered instead, among other statistics: (1) folder count, (2) file count, files in root count (would also count /proc and /sys files which we may not want to include), (3) files list, folder list, (4) frequency vector for each type of file.

We collect the sources file structure and save the data to a parquet file, which is synced in our processed data folder under the project's main data set. Parquet is an open-source file format for Hadoop cluster. Parquet stores nested data structures in a flat columnar format. We chose this format, because by storing data of the same type in each column, we can use encoding better suited to the modern processors' pipeline by making instruction branching more predictable.

3.3. Data Collection System Architecture

The current architecture relies heavily on Amazon Web Services (AWS), since the original arXiv bulk data is stored on AWS S3 (Alexandru *et al*., 2019), (Baboi *et al*., 2019). Given that we only synchronize data from the arXiv buckets, we run a spot instance job. The Proposed Architecture (see Fig. 1) for the current process is:

- **Step 1** – bulk data is synchronized to our S3;

- **Step 2** – we initialize a spot instance;
- **Steps 3 & 4** – we batch run a notebook in order to decompress acquired data to the desired level, build manifests and statistics of processed files, save into parquet files;
- **Step 5** – parquet files are saved on S3.

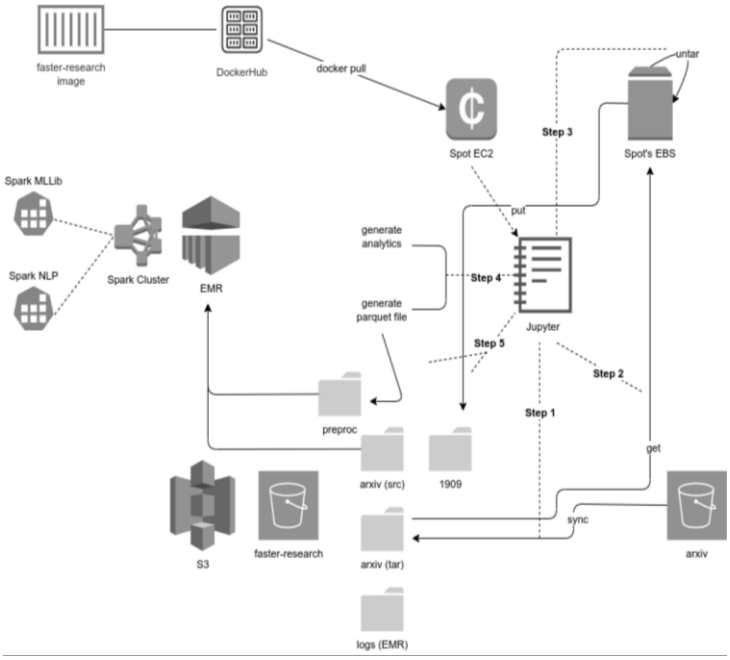


Figure 1: Data Collection System Architecture

If we were to keep everything under the AWS platform, we would consider using EMR (Elastic MapReduce) / Spark NLP[4] / Spark MLLib (MLlib is Spark's machine learning library), as shown in Fig. 1.

4. Statistics and Interpretation

4.1. Analysis

Fig. 2 shows how to use different articles in PDF format, which we found in the arXiv platform.

Based on the data collected from the OAI-PMH interface of arXiv, we have checked some general trends regarding current research under the cs.AI subject category, which is a subcategory and/or leaf node according to the subject categories tree in arXiv. Some of the hierarchical structure is reflected in the data - there will seldom be references of cs.CL (Computational Linguistics) without cs.AI, ignoring certain associations that would be intuitive, like cs.AI - cs.LG - stat.ML (Machine Learning under Computer Science, Statistics respectively).

Based on Fig. 3, the number of collaborators of a paper has increased year by year; while papers in the '90s or '00s had 2 collaborators on average and a maximum of 4-5

[4] Spark NLP is a Natural Language Processing library built on top of Apache Spark ML.

collaborators, in the past couple of years the upper threshold has increased significantly to 8 collaborators per paper. It is probable that these trends would be more transparent if we looked into specific conference/venue data available through graphs such as Open Review (OpenReviewTeam, 2018), Open Academic Graph (Tang *et al.*, 2008, Sinha *et al.*, 2015), or SciGraph (Springer Nature SciGraph).

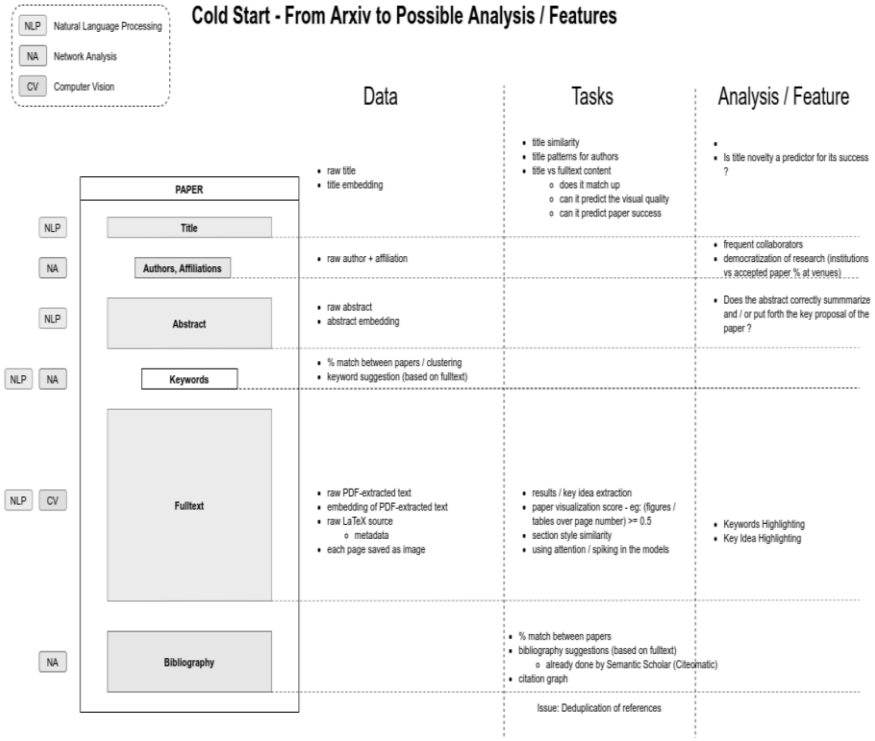


Figure 2: Various ways to use full text PDFs

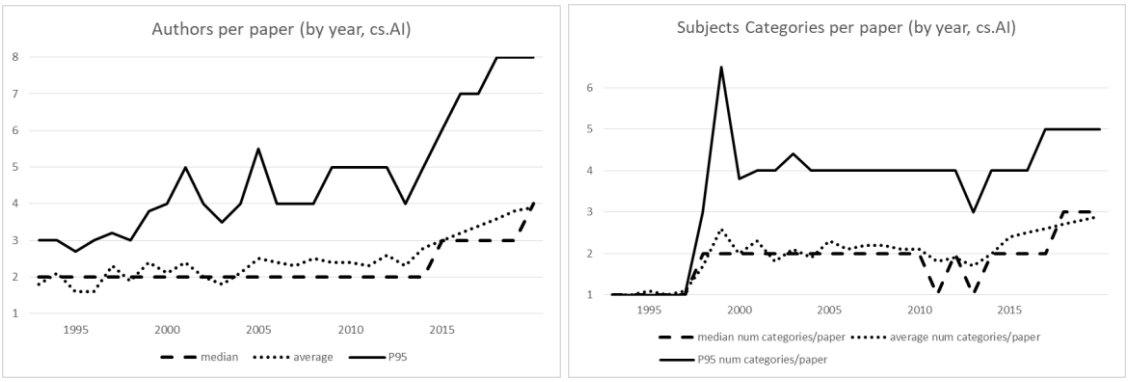


Figure 3: Most papers will fit under cs.AI and 1-3 other categories.

According to Fig. 4, there seems to be some discrepancy in the gathered arXiv metadata. For example, in 2000, 30.58k papers sources have been published (2.75 GB file dump), while only 25k papers have the full text published (6.51 GB file dump), which, regardless of the data size, is contrary to the policy observed in cleaning sources bucket data - if the paper does not have sources uploaded, it will have a compressed full

text instead and if a paper is published on arXiv, it will first have at least full text, sources being optional.

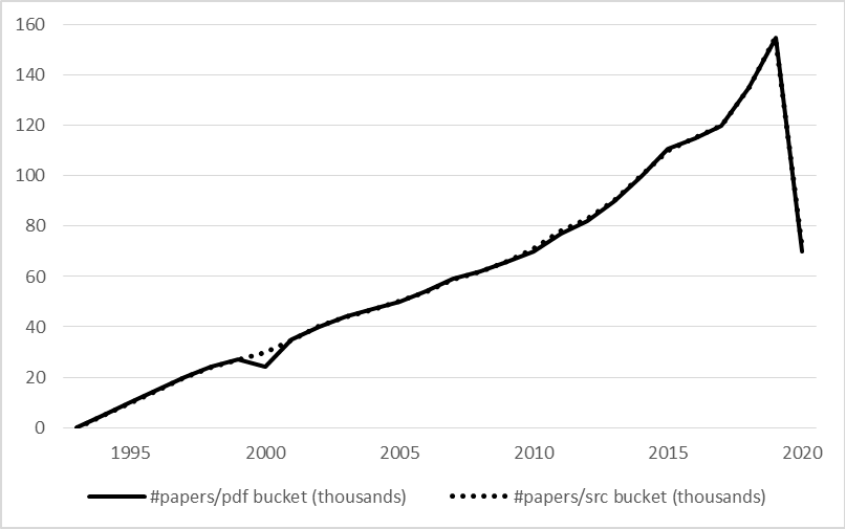


Figure 4: Paper counts (line) and paper size (hover) for PDF and source buckets - aggregated by year - the latest peak being the complete 2019 data with 156,000 papers in total of 239 GB in LaTeX sources and 246 GB in full text PDFs

Another discrepancy is given by the actual size of arXiv bulk data: the official "arXiv Bulk Data Access - Amazon S3" documentation only specifies a ballpark figure of 270 GB PDF and 190 GB sources at 2012-02 (which is about 10GB more for the PDF bucket size). Full arXiv Data (1991 - 2020-06) amounts to 1.46 TB PDF, 1.3 TB sources. Examples keys for these files with the arXiv bucket are: "pdf/arXiv_pdf_2005_035.tar" and "src/arXiv_src_2005_036.tar".

Another note is that arXiv retroactively updates files (if manifests are correct): *e.g.*, pdf/arXiv_pdf_2003_045.tar, which should have last been updated late March - early April, was last updated on 2020-06-07. This complicates the task of keeping an up-to-date copy of all arXiv data and it is unclear whether this happens because of papers being added retroactively or because of papers that are updated (new versions).

According to the AWS calculator, holding all arXiv Bulk Data will cost approx. 65.32 USD monthly just to store (784 USD yearly). This is only to hold the compressed tar's, a TCO (Total Cost of Ownership) being hard to estimate for the moment (AWS EMR usage can also increase the cost by a couple of factors), but would include:

- the above data uncompressed (at least to a paper-level compression, which can be left compressed if the compute time vs. storage cost yields cheaper for compute time) - for both full-text pdf and sources after joining;
- source files manifests, as obtained through ETL Notebook.ipynb;
- textual content of full text, with/out formulas or figure captions, as obtained through PDF to TXT, such as in pdf_txt_utils.py;

- each paper page as an image, as obtained through PDF to JPEG, such as in pdf_img_utils.py - depending on the image format chosen, this can lead to the biggest increase in size amongst augmented data;
- text/image embeddings, classifications, clusters and/or any other attributes obtained through the developed tasks under modules-py/ (*e.g.*, summaries of each paper, obtained through various methods).

4.2. Summarization

In the first phase, we have performed summarization of full text papers using traditional methods such as: Luhn, Edmundson, LSA, LexRank, TextRank, along with some derivate methods. Based on multiple tests, we have found the following (examples in Table 1):

- Luhn does not help much with our dataset, after a point it can give a lot of numbers extracted from the dataset;
- Edmundson gives similar results, but might be better after an NLP-powered selection of bonus/stigma word lists;
- LSA gives one of the best results currently;
- LexRank tends to extract snippets such as theorems;
- TextRank has results on par or better than LSA, even with more technical papers which would involve more formulas/theorems;
- KLSummarizer is very inefficient, most other methods mentioned here run on a 17 page paper from 300ms (Luhn, Edmundson) to 4s (TextRank), but KLSummarizer can run for 2m20; the implementation is based on (Haghighi and Vanderwende, 2009) and "greedily adds sentences to a summary so long as it decreases the Kullback–Leibler Divergence"[5].

Table 1: Sample summaries with the aforementioned methods. While the approximatively 300-character limit here is very low, we did this in interest of space.

Method	Time	Summary
Luhn	250ms	*"The ongoing vulnerability of classifiers was highlighted in recent work by Athalye et al. (2018) and Athalye & Sutskever (2017) that broke an entire suite of defences presented in ICLR 2018 including thermometer encoding (Buckman et al., 2018), detection using local intrinsic dimensionality (Ma et al. 2018])."*
LSA	1.11s	*"[...similar to Luhn...] Fortunately, researchers have been able to derive "algebraic" isoperimetric inequalities that provide lower bounds on the size of the eps-expansion of sets without identifying the shape that achieves this minimum (Talagrand, 1996; Milman & Schechtman, 1986). This also shows that for big MNIST, as the [...]"*
LexRank	2s	*"Theorem 2 (Adversarial examples on the cube). When adversarial examples are defined in the L2-norm (or for any p*

[5] KL Sum algorithm for text summarization (opengenus.org)

		>= 2 the bound in equation 4 becomes 1 - Uc exp(-pi"
TextRank	4.29s	*"The robustness achieved on more complicated datasets such as CIFAR-10 and ImageNet are nowhere near that of MNIST, which leads some researchers to speculate that adversarial defence is fundamentally harder in higher dimensions – an issue we address in Section 8. original Linf-norm L0-norm (sparse) spa[...]"*
KLSummarizer	2m23s	*"The m object classes are defined by probability density functions m c=1, A "random" point from class c is a random variable with density c. 0.0 0.2 0.4 0.6 0.8 1.0 0.4 0.5 0.6 0.7 0.8 0.9 1.0 measure dimension=100 experiment theory 0.0 0.2 0.4 0.6 [...]"*

4.3. Logging

Since most initial work is unsupervised, multiple approaches have to be used and doing this on the whole dataset might be costly, so initial runs will need extensive profiling.

Currently, each function call is logged with the parameters it was called with, time it took to run and errors/exceptions in a CSV format. Each module will initialize its own instance of the logger, which allows for usage as both decorator and custom logging by each function.

The sole issue that might come with logging is that it might also catch logging messages from used libraries (at least gensim does this), which leads to an issue in having a normalized CSV, but can be filtered out in the analysis phase (or the logging module can be set at another debug level, though this is just a partial solution; libraries might use said logging level in exceptional situations).

Malformed or broken log messages should be handled fine at the moment, given each object is stringified, unicode escaped and sliced to an established maximum string threshold (currently, LEN_STR_TRESH = 50). Logging each function argument (*args, kwargs*) and estimating length/size is non-trivial to do automatically (given the variety of function inputs that might not have *len* and/or *getsizeof*), but will be done by custom logging for every function where it may present interest (*e.g.*, PDF size on disk, extracted txt length, passed object size).

5. Conclusions

Given the goal of this project, we have focused mostly on building a collection of datasets and an NLP pipeline:

- raw data has been collected, along with sources and potential new data (under data/raw/);
- Cleaned and pre-processed (under data/preproc/), but also enhanced (extracted text/images, summarized, etc.);
- analysed in Jupiter notebooks (under notebooks/).

We have shown that disparate datasets can be combined in order to allow for processes that have been previously manually executed, for the most part. We are confident that research will be more streamlined in the near future, reviewing will be aided by

automatic tools and researchers everywhere will have more access to current research and will cope more easily with the increased quantity and velocity of research.

Acknowledgements

This work was supported by project REVERT (taRgeted thErapy for adVanced colorEctal canceR paTients), Grant Agreement number: 848098, H2020-SC1-BHC-2018-2020/ H2020-SC1-2019-Two-Stage-RTD.

References

Alexandru, D., Iftene, A. and Gîfu, D. (2019). Using New Technologies to Learn Programming Languages. In *28th International Conference on Information Systems Development (ISD2019)*, August 28-30, Toulon, France.

Athalye, A. and Sutskever, I. (2017). Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*.

Athalye, A., Carlini, N. and Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*.

Baboi, M., Iftene, A. and Gîfu, D. (2019). Dynamic Microservices to Create Scalable and Fault Tolerance Architecture. In *23rd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems*. Procedia Computer Science, 4-6 September, Budapest, Hungary, 159, 1035-1044.

Bensman, S.J., Daugherty, A., Smolinsky, L.J., Sage, D.S. and Katz, J.S. (2014). Power-law distributions, the h-index, and Google Scholar (GS) citations: a test of their relationship with economics Nobelists. *arXiv preprint arXiv:1411.0928*.

Buckman, J., Roy, A., Raffel, C. and Goodfellow, I. (2018). Thermometer Encoding: One Hot Way To Resist Adversarial Examples, in *Proceedings of International Conference on Learning Representations, ICLR* 2018.

Ciaccio, E.J., Bhagat, G., Lebwohl, B., Lewis, S.K., Ciacci, C. and Green, P.H. (2019). Comparison of several author indices for gauging academic productivity. *Informatics in Medicine Unlocked*, 15, 100166, https://doi.org/10.1016/j.imu.2019.100166

Cioffi, R., Travaglioni, M., Piscitelli, G., Petrillo, A. and De Felice, F. (2020). Artificial Intelligence and Machine Learning Applications in Smart Production: Progress, Trends, and Directions. In *Susitanability*, 12, 492; doi:10.3390/su12020492.

Gifu, D. and Teodorescu, M. (2014). Communication Concepts vs. Sciences Concepts. *International Letters of Social and Humanistic Sciences*, 29, 48-57.

Gschwend, T. (2005). Analyzing Quota Sample Data and the Peer-Review Process. *French Politics*. 3. 10.1057/palgrave.fp.8200068.

Gupta, N.S. (2017). Literature Survey on Artificial Intelligence. Available online: https://www.ijert.org/research/a-literature-survey-on-artificial-intelligence-IJERT CONV5IS19015.pdf (accessed on 7 January 2020).

Henderson, P. and Brunskill, E. (2018). Distilling Information from a Flood: A Possibility for the Use of Meta-Analysis and Systematic Review in Machine Learning Research. In *CoRR*, http://arxiv.org/abs/1812.01074.

Horzyk, A. (2014). P-INDEX-a fair alternative to h-index, Department of Automatics and Biomedical Engineering. Poland. http://home.agh.edu.pl/~ horzyk/papers/P-index.pdf.

Klein, M., Broadwell, P., Farb, S. and Grappone, T. (2019). Comparing Published Scientific Journal Articles to Their Pre-print Versions. *International Journal on Digital Libraries*, 20(4), 335-350.

Lee, Y.H. (2018). An overview of meta-analysis for clinicians. *The Korean Journal of Internal Medicine* 33 (2018), 277-283.

Lee, P.-S., West, J.D. and Howe, B. (2018). Viziometrics: Analyzing Visual Information in the Scientific Literature. *IEEE Transactions on Big Data* 4, 117-129.

Ma, X., Li, B., Wang, Y, Erfani, S.M., Wijewickrema, S., Schoenebeck, G., Song, D., Houle, M.E. and Bailey, J. (2018). Characterizing Adversarial Subspaces using Local Intrinsic Dimensionality, *arXiv preprint arXiv:1801.02613*, https://arxiv.org/pdf/1801.02613.pdf.

Milman, V.D. and Schechtman, G. (1986). Asymptotic Theory of Finite Dimensional Normed Spaces. *Lecture notes in mathematics*, 1200, Springer-Verlag, Berlin, Heidelberg, 1-156.

Sinha, A., Shen, Z., Song, Y., Ma, H., Eide, D., Hsu, B.J. and Wang, K (2015). An Overview of Microsoft Academic Service (MAS) and Applications. In *Proceedings of the 24th International Conference on World Wide Web. WWW '15 Companion*. Florence, Italy: Association for Computing Machinery, 2015, 243-246.

Szklo, M. (2006). Quality of Scientific Articles. *Revista de Saúde Pública*, 40, 30-35.

Talagrand, M. (1999). A new look at independence. *The Annals of probability*, 1-34.

Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L. and Su, Z. (2008). ArnetMiner: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 990-998.

Waltman, L. and Jan van Eck, N. (2012). The Inconsistency of the h-index. In *Journal of the American Society for Information Science and Technology*, 63(2), 406-415.

Wickson, F. and Carew, A.L. (2014). Quality Criteria and Indicators for Responsible Research and Innovation: Learning from Transdisciplinarity. *Journal of Responsible Innovation*, 1(3), 254-273.

Yokuş, G. and Akdağ, H. (2019). Identifying Quality Criteria of a Scientific Research Adopted by Academic Community: A Case Study. *International Journal of Eurasia Social Sciences*. 10, 516-527.

INDEX OF AUTHORS