Input Encoding for Sequence-to-Sequence Learning of Romanian Grapheme-to-Phoneme Conversion

Adriana STAN
Communications Department
Technical University of Cluj-Napoca, Romania
adriana.stan@com.utcluj.ro

***Abstract*—This paper evaluates the use of sequence-to-sequence learning models for the Romanian grapheme-to-phoneme conversion. The strategies explore the use of different input feature encoding: one-hot letter encoding, additional embedding layer and grapheme embeddings learned from a large corpus of Romanian text. Additional lexical information, such as syllabification and lexical stress is also taken into consideration for augmenting the orthographic form of the word and providing more accurate phonetic transcriptions.**

The sequence-to-sequence models are also compared to a baseline decision tree algorithm in terms of both phone- and word-level accuracy. The best results are achieved by the model which uses grapheme embeddings and all additional linguistic information. Its accuracy is 97.90% at word-level, and 99.62% at phone-level. However, only minor differences exist between the tested systems.

***Index Terms*—phonetic transcription, Romanian, grapheme-to-phoneme, G2P, LTS, LSTM, DNN, letter embeddings, grapheme embeddings**

I. Introduction

The latest trends in human-computer interaction using natural language or speech processing tend to limit the input feature manipulation by the human expert in favor of a deep learning architecture. This architecture can, at least in theory, extract similar features on its own. For example, state-of-the-art speech synthesis systems are now trained using only pairs of text and audio with no segment alignments or text processing [1], [2]. Machine translation systems can learn to map sequences of texts in two separate languages, without the need of an annotated correspondence between the words of the sequences.

Even though the additional features are not required in these particular tasks, they are still an essential part of many other applications. For example, the lexical stress or phonetic transcription can discriminate between homographs in semantic analysis. To develop algorithms which can extract these features with high accuracy, large lexicons and datasets are required. In this respect, Romanian is an under resourced language [3] and does not benefit from readily available, high quality language resources and systems. However, in recent years more and more resources and tools have been developed and published by the research community.

One of the most important linguistic resources for Romanian was developed within the CoRoLa project [4], [5]. The CoRoLa corpus of Romanian texts contains over 1 billion tokens. The data is cleaned, partially annotated with expert linguistic metadata and includes multiple text styles. The datasets can be accessed through an online application.[1]

The open-source initiative of the online Romanian Explicative Dictionary [6] is also a great resource for extracting word inflections or part-of-speech tags, as well as the syllabic forms of the words which do not follow the general syllabification rules in Romanian.[2]

Scientific publications of Romanian textual resources include: Barbu et al. [7] published two Romanian dictionaries of syllabic and inflected forms for over 500,000 Romanian words of 65,000 different lemmas. Simionescu released an extensive list of part-of-speech annotations for over 1,000,000 words [8].[3] Domokos et al. [9] developed an extensive grapheme-to-phoneme dictionary in SAMPA format, called NAVIRO, and built from an initial list of 10,000 manually transcribed words, and extended using artificial neural networks.[4] Another freely available grapheme-to-phoneme dictionary containing over 75,000 entries composed mainly of the Romanian Scrabble's Association's official list of words was introduced in [10].[5]

With the availability of these resources, other studies and tools have been published, investigating either entire text processing systems or only parts of them. The following listing includes only those works related to the task of grapheme-to-phoneme conversion or complete systems which contain the phonetic conversion among other processes.

In [11] the use of neural networks for grapheme-to-phoneme conversion in the context of text-to-speech synthesis is investigated, and a 98.3% word-level accuracy is reported. A similar approach, based on artificial neural networks, is presented in [12] and [13]. The authors of [13] report a 92.83% accuracy at phone-level. A set of rules for the Romanian grapheme-to-phoneme transcription was developed in [14] and obtained 95% accuracy. Similar rule-based transcriptions combined with decision trees are explored in [15]. The accuracy of the method at word-level is 94.8% on one of the evaluation subsets. The authors of [16] compare 5 separate methods based on: decision trees, neural networks, support vector machines,

[1] http://corola.racai.ro/
[2] http://dexonline.ro
[3] http://nlptools.infoiasi.ro/WebPosTagger
[4] http://users.utcluj.ro/~jdomokos/naviro/
[5] http://speech.utcluj.ro/marephor/

pronunciation by analogy and an expert system. Their best results have an accuracy of 96.68% at word-level.

The work in [17] achieves a 93% word-level accuracy for the Romanian grapheme-to-phoneme conversion using a maximum entropy classifier and a custom data-driven algorithm. [18] introduces a margin infused relaxed method and a specialized algorithm for same processing task. The reported word-level accuracy is 96.29%. The same authors present in [19] a series of decision tree-based evaluations for multiple Romanian and English text processing tasks, including grapheme-to-phoneme conversion. The word-level accuracy of the G2P module was reported at 95.05%.

A novel approach to the phonetic transcription task was presented in [20] where grapheme-to-phoneme conversion is performed using statistical machine translation principles and obtains a 97.24% accuracy at word-level. The authors of [10] use decision trees with various context lengths and achieve a 99.61% accuracy at phone-level. The decision trees are compared with deep learning networks in [21]. The best algorithm obtained at most 99.63% accuracy at phone-level.

Table I summarizes these works and their reported accuracy. However, the methods are not directly comparable as the training datasets are not consistent across the studies. Also, the level of the reported accuracy (i.e. phone- or word-level) differs.

It is also worth mentioning a few papers which introduced full text processing systems, such as [22] which addresses the diacritic restoration, text normalization, syllabification, phonetic transcription and lexical stress positioning; [23] describes a complete text-to-speech synthesis system including the front-end text processing with syllabification, lexical stress assignment and grapheme-to-phoneme conversion. [24] presents the authors' work on a full text processing tool for phonetic transcription, syllabification and part-of-speech tagging.

In terms of grapheme-to-phoneme conversion for other languages, the most recent approaches make use of the complex deep learning architectures. Studies on this topic are numerous and include various mono- or multi-lingual tasks, as well as the use of unsupervised representation learning for the graphemes: [25] introduces a bidirectional long short-term memory architecture combined with alignment-based models for translating English words into their phonetic representations. [26] adapts good G2P models for low-resource languages in a multilingual framework. [27] experiments with uni- and bi-directional LSTMs with various output delays combined with an n-gram language model. [28] uses a multitask learning strategy combined with n-gram language models to improve the G2P of English and German texts. [29] learns global character vectors from plain text resources and applies them to monolingual and multilingual G2P conversion in a recurrent neural network setup.

TABLE I
RESULTS REPORTED IN THE LITERATURE FOR THE TASK OF GRAPHEME-TO-PHONEME CONVERSION IN ROMANIAN

Paper	Reference	Level	Accuracy
(Burileanu, 2002)	[11]	word	98.30%
(Ordean et al., 2009)	[15]	word	94.80%
(Toma et al., 2009)	[14]	word	95.00%
(Domokos et al., 2011)	[13]	phone	92.83%
(Toma et al., 2013)	[16]	word	96.68%
(Boroş et al.,2012)	[17]	word	93.00%
(Boroş et al., 2013)	[18]	word	96.29%
(Cucu et al., 2014)	[20]	word	97.24%
(Boroş et al. , 2017)	[19]	word	95.05%
(Toma et al., 2017)	[10]	phone	99.61%
(Stan et. al, 2018)	[21]	phone	99.63%

Starting from this overview, the aim of this paper is to investigate the use of the newly developed sequence-to-sequence deep learning models [30] applied to the Romanian grapheme-to-phoneme conversion. The method evaluates a simple encoder-decoder structure with different grapheme input encoding and compares the results with those of a basic decision tree algorithm.

The paper is organized as follows: Section II describes the sequence-to-sequence algorithm and its internal network structure. Section III introduces the datasets and evaluation procedures, while conclusions and discussions are presented in Section IV.

II. SEQUENCE-TO-SEQUENCE LEARNING FOR GRAPHEME-TO-PHONEME CONVERSION

The task of grapheme-to-phoneme conversion implies variable length input and output sequences, i.e. the length of the word versus its phonetic transcription. The recently introduced sequence-to-sequence models [30] can do just that. Their structure involves an *encoder-decoder* architecture. The encoder aims to create a low-dimensional representation of the input sequence which captures the essential information for the task at hand. The output of the encoder is then used to condition the decoder's output. The decoder is trained on time-delayed sequences, meaning that it learns to predict the next token of the sequence. Both the encoder and decoder are neural networks with recurrent or convolutional structures [31].

This work uses the recurrent architecture. Recurrent neural networks (RNN) are a class of neural networks in which connections between the nodes are made so that temporal sequences can be accurately modeled [32]. This means that the output at time step t_i is conditioned on the state of the network at time step t_{i-1} along with the current input.

However, in vanilla RNNs, sequences with long temporal dependencies cannot be represented properly. The solution to this problem is to use a specialized type of neural node able to model longer temporal dependencies in the input data. One such node is the Long Short Term Memory (LSTM) [33]. LSTMs have a more complex internal structure (see Figure 1) which includes a series of gates designed to either allow, partially allow or block the current information to pass to the next time step. LSTMs have been successfully applied to several complex tasks in NLP, such as machine translation [30], language modeling [34] or text generation [35].

The equations describing the behavior of the LSTM are:

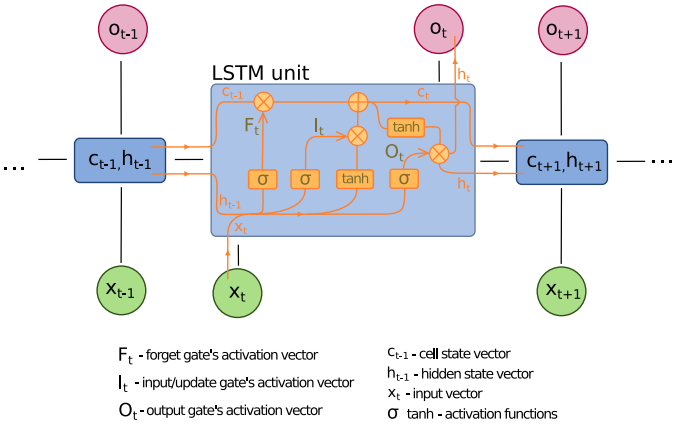

Fig. 1. Diagram for a one-unit Long Short-Term Memory (LSTM) [36]

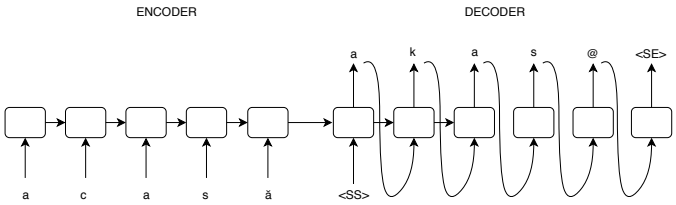

Fig. 2. Sequence-to-sequence model architecture. <SS> and <SE> mark the sequence-start and sequence-end tokens, respectively.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{1}$$

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{3}$$

$$c_t = f_t C_{t-1} + i_t \tanh\left(W_c[h_{t-1}, x_t] + b_c\right) \tag{4}$$

$$h_t = o_t \tanh(c_t) \tag{5}$$

where σ and $tanh$ are the sigmoid and hyperbolic tangent functions, respectively. x_t is the input vector to the LSTM unit, i_t is the input/update gate's activation vector, f_t is the forget gate's activation vector, and o_t is the output gate's activation vector at time step t. W and b are the corresponding weight matrices and bias vectors for each gate–these parameters are learned in the training process. c_t is the cell state vector at time step t, and h_t is the hidden state vector of the LSTM unit. h_t is also known as the output vector of this cell type.

Across all experiments the same sequence-to-sequence model architecture was used. The architecture is composed of a single layer LSTM in both the encoder and the decoder, as shown in Figure 3. However, several input sequence encoding strategies were evaluated: *one-hot encoding*, *embedding layer* and *grapheme embeddings*. All encoding was performed at *grapheme-level*.

The **one-hot encoding (OHE)** simply creates a 2D array with the dimension MxN, where M is equal to the maximum length of the input sequences, and N is equal to the number of distinct characters available in the input sequences. Each

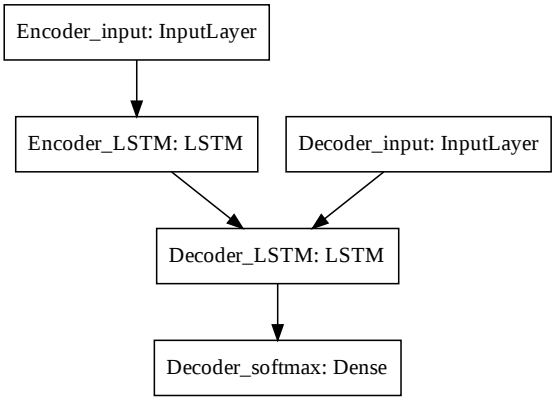

Fig. 3. Sequence-to-sequence network architecture

row has null elements except for one which is equal to 1. The index i of the element equal to 1 is determined as the position of the current input character c in the set of all possible input characters $\{c_1, c_2, ..., c_i, ..., c_N\}$. For example, if the set of all possible input characters is $\{a, b, c\}$ and we want to encode the sequence $aababc$ the OHE of this sequence is:

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The **embedding layer (EL)** is an additional neural layer added between the encoder's input and the LSTM. The embedding layer is jointly learned with the rest of the sequence-to-sequence model's parameters. The task of the embedding layer is to convert the input characters in each sequence into dense vectors of fixed-length. The dense vectors should provide better approximations of the input characters' relevance to the given learning task.

However, the training data in most NLP applications is rather limited. Therefore, a fixed-length embedding of the input characters could be learned from an external text resource, such as large amounts of raw texts. This embedding would approximate the context in which each character from the set can be found in a particular language. This **grapheme embedding (GE)** is similar to the word embeddings [37]. The difference is that vectors are obtained at character/letter level. Word embeddings have proved their efficiency in a multitude of NLP tasks [38], and grapheme embeddings were shown to mimic the behavior of the phonetic transcription [39].

TABLE II
EXAMPLE OF INPUT-OUTPUT SEQUENCES FOR THE WORD *acasă* (EN. *home*) USING THE ORTHOGRAPHIC FORM, ORTHOGRAPHIC PLUS SYLLABIFICATION, ORTHOGRAPHIC PLUS LEXICAL STRESS, AND ALL 3 COMBINED.

Type	Input seq	Output seq
Orthographic	acasă	a k a s @
Orthographic+Syllabification	a-ca-să	a k a s @
Orthographic+Lexical stress	ac'asă	a k a s @
Orthographic+Syllabification+Lexical stress	a-c'a-să	a k a s @

III. EVALUATION

A. Training data

To evaluate the performance of sequence-to-sequence models applied to Romanian grapheme-to-phoneme conversion, the MaRePhor [10] phonetic dictionary was used.[6] The dictionary consists of 72,375 words and 591,570 letters. The entries are words from the Romanian Scrabble Association's official list of words and the entries from a 15,517 words dictionary developed according to the SpeechDat specifications. The phonetic transcriptions are in SAMPA format.[7]

Aside from the direct grapheme-to-phoneme conversion, additional linguistic information was added to the input data with the aim of aiding the phonetic transcription. Syllabification and lexical stress of the words were appended to the input features. The syllabification of approximately 507,000 words was extracted from the RoSyllabiDict lexicon [7]. The DEX Online Database [6] which includes over 1,600,000 words and their inflected forms along with the stress labels was selected for lexical stress assignment.

Combining the common words in all 3 dictionaries a list of 62,874 words was obtained. The syllabification and lexical stress were incorporated into the orthographic form of the word, either individually or simultaneously. The output sequence was in all experiments only the phonetic transcription. Table II shows an example of the entries. The data was randomly split into training (80%) and testing (20%) sets. The split was maintained across all evaluations so that there are no differences between the systems caused by the random split of the training and testing datasets.

The Wikipedia database dump for Romanian [40] was processed through a word2vec model using the Gensim toolkit[8] to obtain the grapheme embeddings. The order of the embedding was to set to 30 so that it is close to the number of letters in the Romanian alphabet (i.e. 31 letters).

A plot of the t-SNE [41] visualizations of the grapheme embeddings is presented in Figure 4. It can be noticed that the vowels *a, e, i, o, u* are closely grouped together. The same grouping can be found for the least frequent letters in Romanian *k, w, y, q*. This means that, at least in theory, the network can make use of better representations for its inputs.

[6]https://speech.utcluj.ro/marephor/
[7]https://www.phon.ucl.ac.uk/home/sampa/romanian.htm
[8]https://radimrehurek.com/gensim/

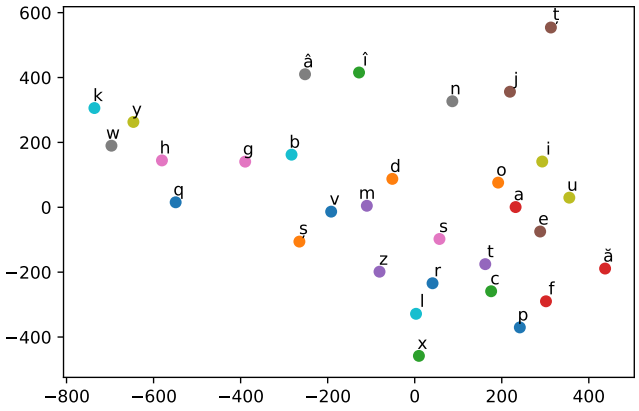

Fig. 4. Bidimensional t-SNE plot of the Wikipedia-based Romanian grapheme embeddings

B. Results

The neural sequence-to-sequence architecture for the Romanian grapheme-to-phoneme task, was implemented using the Keras[9] toolkit with TensorFlow backend.[10] Because of the non-sequential structure of the network, the functional API of Keras was used. This enables us to merge the input of the decoder with the hidden state of the encoder.

As a first step in the evaluation, we explored the hyperparameter setup of the network. Initial tests were carried out for the dimension of the batch size and the latent dimension of the LSTM layers in the encoder and decoder. The best results were obtained using a batch size of 32 and a latent dimension of 512 nodes. The embedding layer consists of 30 nodes so that there is a correspondence between the externally-learned grapheme embeddings and this one. The weights of the network were optimized using RMSprop, an algorithm similar to Adagrad [42] which tunes the learning rate depending on an running average of the recent gradients. The loss function was set to categorical cross-entropy due to the multiclass output of the decoder. The number of epochs was set to 20 for the OHE and GE encodings. For the EL encoding, because the network has to learn additional weights, the number of epochs was set to 40.[11].

Accuracy results of the 3 sequence-to-sequence networks with their respective input embeddings are presented in Tables III, IV and V. The accuracy is measured at phone- and word-level. Phone-level accuracy took into account all the predicted phones, and not just the ones that pose problems in Romanian (see [10]). It can be noticed that the GE input achieves the highest accuracy scores: **99.51%** at phone-level and **97.40%** at word-level when considering only the orthographic form of the word as input. If the syllabification and

[9]https://keras.io/
[10]https://www.tensorflow.org/
[11]Jupyter notebooks for all systems' training flow are available here: http://github.com/speech-utcluj/

TABLE III
ACCURACY RESULTS FOR **SEQ2SEQ** WITH **ONE-HOT ENCODING**

Input features	Accuracy [%] Phone	Word
Orthographic	99.42	97.05
Orthographic+Syllabification	99.12	96.40
Orthographic+Lexical stress	99.01	95.85
Orthographic+Syllabification+Lexical stress	99.40	97.45

TABLE IV
ACCURACY RESULTS FOR **SEQ2SEQ** WITH **WIKI-BASED GRAPHEME EMBEDDINGS**

Input features	Accuracy [%] Phone	Word
Orthographic	99.51	97.40
Orthographic+Syllabification	99.30	96.40
Orthographic+Lexical stress	99.56	97.70
Orthographic+Syllabification+Lexical stress	**99.62**	**97.90**

TABLE V
ACCURACY RESULTS FOR **SEQ2SEQ** WITH **EMBEDDING LAYER**

Input features	Accuracy [%] Phone	Word
Orthographic	98.75	95.40
Orthographic+Syllabification	98.42	95.26
Orthographic+Lexical stress	99.01	96.00
Orthographic+Syllabification+Lexical stress	99.15	96.25

TABLE VI
ACCURACY RESULTS OF THE **DECISION TREE CLASSIFIER** FOR A WINDOW LENGTH OF **5 CHARACTERS**

Input features	Accuracy [%] Phone	Word
Orthographic	99.54	96.71
Orthographic+Syllabification	99.52	96.68
Orthographic+Lexical stress	99.56	96.95
Orthographic+Syllabification+Lexical stress	**99.60**	**97.17**

the lexical stress are added to the input features, the accuracy increases to **99.62%** at phone-level and to **97.90%** at word level. Although there results are not directly comparable to the ones presented in Table I due to the different training sets. It is, however, safe to state that the sequence-to-sequence method achieves similar accuracies as the state-of-the-art, if not higher.

However, the difference between the GE and OHE embeddings are not statistically significant. This can be explained by the simple letter-to-sound rules in Romanian which can be easily learned by this complex structure alone. It might be the case that for languages where the grapheme-to-phoneme conversion poses more complex problems, this difference could increase. The EL has slightly lower accuracy values, despite using twice the number of training epochs. This result can be explained by the fact that the EL might need more training data for the additional weights of the network. It is also interesting to note the fact that the syllabification and the lexical stress do not add that much value to the output accuracy. And also that the syllabification alone can reduce it. Again, it might be valuable to apply the same strategy to a more complex G2P language, as the Romanian results might already be plateaued.

As baseline, a simple decision tree classifier was also evaluated. The input to the decision tree is a window of graphemes centered around the predicted grapheme. Results for this algorithm are presented in Table VI for phone- and word-level accuracies, and with the addition of the linguistic information, i.e. syllabification and lexical stress. The feature encoding for the decision tree follows the steps described in [21]. Results show that even with this simple algorithm, the G2P task for Romanian can be solved with rather high accuracy.

IV. CONCLUSIONS

This paper evaluated the use of sequence-to-sequence learning strategies for Romanian grapheme-to-phoneme conversion. The strategies involved the use of various input feature encodings, such as one-hot encoding, an additional embedding layer, or grapheme embeddings learned from an external text resource. The evaluation also included an analysis of combining the orthographic form of the words with their syllabification, lexical stress or both. The results of the systems were compared in terms of phone- and word-level accuracy scores. A decision tree algorithm trained to predict each phone individually from a character window sequence was included as baseline. The best results were obtained with the grapheme embeddings and all additional linguistic information, and achieve a phone-level accuracy of 99.62% and word-level accuracy of 97.90%. However, the difference between all the setups is not significant, and means that the sequence-to-sequence strategy is sufficient for the G2P task in Romanian.

One problem noticed in the sequence-to-sequence model prediction is that, because the decoder is conditioned only on the hidden state of the encoder, and that it only learns to predict the next phoneme in the output sequence, sometimes the order of the output phonemes is scrambled. One way to overcome this issue would be to condition the decoder on the current input grapheme as well, or to extend the available training data.

As future work, other network architectures could be considered. But it would also be interesting to test the simultaneous prediction of the phonetic transcription, syllabification and lexical stress assignment.

ACKNOWLEDGMENT

This work was supported by a grant of the Romanian Ministry of Research and Innovation, PCCDI – UEFISCDI, project number PN-III-P1-1.2-PCCDI-2017-0818/73, within PNCDI III.

REFERENCES

[1] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous, "Tacotron: Towards End-to-End Speech Synthesis," in *Proc. Interspeech*, 2017.

[2] W. Ping, K. Peng, A. Gibiansky, S. Ö. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller, "Deep Voice 3: 2000-Speaker Neural Text-to-Speech," *CoRR*, vol. abs/1710.07654, 2017.

[3] D. Trandabat, E. Irimia, V. Barbu-Mititelu, D. Cristea, and D. Tufis, "The Romanian Language in the Digital Era," *Springer, Metanet White Paper Series*, 2012.

[4] V. B. Mititelu, E. Irimia, and D. Tufis, "CoRoLa:The Reference Corpus of Contemporary Romanian Language," in *LREC*, 2014, pp. 1235–1239.

[5] D. Tufiș, V. B. Mititelu, E. Irimia, Ș. D. Dumitrescu, and T. Boroș, "The IPR-cleared corpus of contemporary written and spoken Romanian language," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 2516–2521.

[6] The Romanian Explicative Dictionary (DEX) online. [Online]. Available: www.dexonline.ro

[7] A.-M. Barbu, "Romanian lexical data bases: Inflected and syllabic forms dictionaries." in *Proceedings of the International Conference on Language Resources and Evaluation*, 01 2008.

[8] R. Simionescu, "Graphical grammar studio as a constraint grammar solution for part of speech tagging," in *The Conference on Linguistic Resources and Instruments for Romanian Language Processing*, vol. 152, 2011.

[9] J. Domokos, O. Buza, and G. Toderean, "100k+ words, machine-readable, pronunciation dictionary for the romanian language," *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pp. 320–324, 2012.

[10] S.-A. Toma, A. Stan, M.-L. Pura, and T. Barsan, "MaRePhoR - An Open Access Machine-Readable Phonetic Dictionary for Romanian," in *Proceedings of the 9th Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, Bucharest, Romania, July, 6-9 2017.

[11] D. Burileanu, "Basic Research and Implementation Decisions for a Text-to-speech Synthesis System in Romanian," *International Journal of Speech Technology*, no. 5, pp. 211–225, 2002.

[12] D. Jitca, H. Teodorescu, V. Apopei, and F. Grigoras, "An ANN-based method to improve the phonetic transcription and prosody modules of a TTS system for the Romanian language," 01 2003.

[13] D. József, B. Ovidiu, and T. Gavril, "Automated grapheme-to-phoneme conversion system for Romanian," in *2011 6th Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, May 2011, pp. 1–6.

[14] T. Stefan-Adrian and M. Doru-Petru, "Rule-Based Automatic Phonetic Transcription for the Romanian Language," in *2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns*, Nov 2009, pp. 682–686.

[15] M. A. Ordean, A. Saupe, M. Ordean, M. Duma, and G. C. Silaghi, "Enhanced Rule-Based Phonetic Transcription for the Romanian Language," in *2009 11th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, Sep. 2009, pp. 401–406.

[16] Ș. Toma, T. Birsan, F. Totir, and E. Oancea, "On letter to sound conversion for Romanian: A comparison of five algorithms," in *2013 7th Conference on Speech Technology and Human - Computer Dialogue (SpeD)*, Oct 2013, pp. 1–6.

[17] T. Boros, D. Stefanescu, and R. Ion, "Bermuda, a data-driven tool for phonetic transcription of words," in *Natural Language Processing for Improving Textual Accessibility (NLP4ITA) Workshop*, 2012.

[18] T. Boroș, "A unified lexical processing framework based on the margin infused relaxed algorithm. a case study on the Romanian language," in *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*. Hissar, Bulgaria: INCOMA Ltd. Shoumen, BULGARIA, Sep. 2013, pp. 91–97.

[19] T. Boros, S. D. Dumitrescu, and S. Pipa, "Fast and accurate decision trees for natural language processing tasks," in *RANLP*, 2017.

[20] H. Cucu, A. Buzo, L. Besacier, and C. Burileanu, "SMT-based ASR domain adaptation methods for under-resourced languages: Application to Romanian," *Speech Communication*, vol. 56, pp. 195 – 212, 2014.

[21] A. Stan and M. Giurgiu, "A Comparison Between Traditional Machine Learning Approaches And Deep Neural Networks For Text Processing In Romanian," in *Proceedings of the 13th International Conference on Linguistic Resources and Tools for Processing Romanian Language (ConsILR)*, Jassy, Romania, November, 22-23 2018.

[22] C. Ungurean and D. Burileanu, "An advanced NLP framework for high-quality Text-to-Speech synthesis," in *2011 6th Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, May 2011, pp. 1–6.

[23] A. Stan, J. Yamagishi, S. King, and M. Aylett, "The Romanian speech synthesis (RSS) corpus: Building a high quality HMM-based speech synthesis system using a high sampling rate," *Speech Communication*, vol. 53, no. 3, pp. 442–450, 2011.

[24] T. Boros, S. D. Dumitrescu, and V. Pais, "Tools and resources for Romanian text-to-speech and speech-to-text applications," *CoRR*, vol. abs/1802.05583, 2018.

[25] K. Yao and G. Zweig, "Sequence-to-sequence neural net models for grapheme-to-phoneme conversion," *CoRR*, vol. abs/1506.00196, 2015.

[26] B. Peters, J. Dehdari, and J. van Genabith, "Massively multilingual neural grapheme-to-phoneme conversion," *CoRR*, vol. abs/1708.01464, 2017.

[27] K. Rao, F. Peng, H. Sak, and F. Beaufays, "Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 4225–4229.

[28] B. Milde, C. Schmidt, and J. Kohler, "Multitask sequence-to-sequence models for grapheme-to-phoneme conversion," in *Proc. Interspeech 2017*, 2017, pp. 2536–2540.

[29] J. Ni, Y. Shiga, and H. Kawai, "Multilingual grapheme-to-phoneme conversion with global character vectors," in *Proc. Interspeech 2018*, 2018, pp. 2823–2827.

[30] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[31] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," *CoRR*, vol. abs/1705.03122, 2017.

[32] Z. C. Lipton, "A critical review of recurrent neural networks for sequence learning," *CoRR*, vol. abs/1506.00019, 2015.

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[34] C. Chelba, M. Norouzi, and S. Bengio, "N-gram language modeling using recurrent neural network estimation," Google, Tech. Rep., 2017.

[35] Z. Xie, "Neural text generation: A practical guide," *CoRR*, vol. abs/1711.09534, 2017.

[36] F. Deloche. Diagram for a one-unit Long Short-Term Memory (LSTM) - Wikimedia Commons. [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/6/63/Long_Short-Term_Memory.svg

[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119.

[38] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu, "Exploring the limits of language modeling," 2016.

[39] O. Watts, "Unsupervised learning for text-to-speech synthesis," Ph.D. dissertation, University of Edinburgh, 2012.

[40] "Linguatools–Romanian Wikipedia database dump," https://linguatools.org/tools/corpora/wikipedia-monolingual-corpora/.

[41] L. van der Maaten and G. E. Hinton, "Visualizing High-Dimensional Data Using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[42] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, Jul. 2011.