INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

RECOApy: Data recording, pre-processing and phonetic transcription for end-to-end speech-based applications

Adriana Stan

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

Abstract

Deep learning enables the development of efficient end-to-end speech processing applications while bypassing the need for expert linguistic and signal processing features. Yet, recent studies show that good quality speech resources and phonetic transcription of the training data can enhance the results of these applications. In this paper, the RECOApy tool is introduced. RECOApy streamlines the steps of data recording and pre-processing required in end-to-end speech-based applications. The tool implements an easy-to-use interface for prompted speech recording, spectrogram and waveform analysis, utterance-level normalisation and silence trimming, as well grapheme-to-phoneme conversion of the prompts in eight languages: Czech, English, French, German, Italian, Polish, Romanian and Spanish.

The grapheme-to-phoneme (G2P) converters are deep neural network (DNN) based architectures trained on lexicons extracted from the Wiktionary online collaborative resource. With the different degree of orthographic transparency, as well as the varying amount of phonetic entries across the languages, the DNN's hyperparameters are optimised with an evolution strategy. The phoneme and word error rates of the resulting G2P converters are presented and discussed. The tool, the processed phonetic lexicons and trained G2P models are made freely available.

Index Terms: speech recording tool, multilingual, phonetic transcription, grapheme-to-phoneme, evolution strategy, sequence-to-sequence, convolutional networks, transformer networks.

1. Introduction

Nowadays, in the development of deep neural networks (DNN) based speech processing applications, most of the signal preprocessing, feature extraction and linguistic annotations are part of the inherent neural learning. This means that systems for automatic speech recognition (ASR) and text-to-speech synthesis (TTS) can be easily trained using only pairs of audio and orthographic transcript [1, 2, 3]. A major advantage of this approach is that training data can be easily and readily found, and that there is no language dependency in the development stage—other than the language specific speech resources. Although this approach yields satisfactory results for most end-user applications, when it comes to high quality systems, found speech data and orthographic input does not suffice [4]. Most of the high-end commercial applications still make use of large amounts of studio recordings and elaborate text processing modules [2, 5].

Hence, there is still a need for tools which can facilitate the development of domain or speaker specific training data, as well as tools which can generate expert linguistic features in a variety of languages. In this context, the first version of the RECOApy tool is introduced. RECOApy was designed with the main purpose of enabling end-users to record their own data and prepare it for end-to-end speech processing applications. It provides an easy to use interface for prompted speech recoding which includes several monitoring and data processing options (see Section 2), as well as a set of highly accurate pre-trained neural network models able to phonetically transcribe the prompts in eight languages.

The task of building grapheme-to-phoneme converters is not novel, but depending on a language's orthographic transparency and onset entropy [6], G2P can be solved using simple rule-based systems (e.g. Finnish) or can pose serious problems even for the most advanced deep learning algorithms (e.g. English). The modern G2P converters aim at solving the problem of phonetic transcription in multiple languages at once. But phonetic lexicons are not readily available in most languages, and researchers are now investigating the use of collaborative online resources, such as Wiktionary,[1] as an alternative. [7] does just this by extracting the phonetic transcriptions in six languages from Wiktionary and validates them over manually crafted lexicons. The authors of [8] also use several online repositories to train and adapt the models from high-resource languages to related low-resource languages. Multilingual G2P was also addressed by changing the grapheme representation: [9] proposes a model which uses byte-level input representation to accommodate different grapheme systems, along with an attention-based Transformer architecture. Ancillary audio data was also used to learn a more optimal intermediate representation of source graphemes in a multi-task training process for multilingual G2P [10].

As the grapheme-to-phoneme task is inherently a sequence to sequence (*seq2seq*) mapping problem, the G2P converter in RECOApy uses this type of learning architecture. Similar approaches were introduced in [11]. The authors map the entire input grapheme sequence to a vector, and then use a recurrent neural network to generate the output sequence conditioned on the encoding vector. [12] describes a G2P model based on a unidirectional LSTM with different output delays and deep bidirectional LSTM with a connectionist temporal classification layer. Milde *et al.* [13] investigate how multitask learning can improve the performance of sequence-to-sequence G2P models. A single seq2seq model is trained on multiple phoneme lexicon datasets containing several languages and phonetic alphabets. Esch *et al.* [14] train recurrent neural network-based models to predict the syllabification and stress patterns of the input text for TTS, while also deriving phonetic transcriptions in the process. The use of entire phrases as input to LSTM, biLSTM and CNN-based neural networks and their evaluation in English, Czech and Russian is presented in [15].

[1] www.wiktionary.org

http://dx.doi.org/10.21437/Interspeech.2020-1184

Starting from this overview of multilingual and neural networks-based training schemes, RECOApy's G2P module incorporates the use of online collaborative phonetic lexicons and lexicon-tailored seq2seq neural network architectures derived with the help of an evolution strategy. The RECOApy tool, along with the parsed lexicons and complete set of trained models are made freely available. The G2P module can be used as a standalone tool as well.

The paper is organised as follows: Section 2 introduces the recording app and its features. Section 3 presents the phonetic transcription tool development and hyperparameter tuning using evolution strategies. Results of the phonetic converters are discussed in Section 3.3, and conclusions are drawn in Section 4.

2. RECOApy GUI

Recording prompted speech by end-users can be easily performed with any of the numerous free general purpose recording tools available, such as Audacity[2] or Wavesurfer [16]. But this means that in order to obtain phrase-length speech segments, the continuous recording stream needs to be manually segmented and aligned to the prompts. Or that the recording operator needs to start and stop the recording after each prompt reading. In both cases incorrect readings need to be marked or deleted. This makes the methods tedious, time consuming and error prone.

RECOApy was developed with the main objective of streamlining the end-user speech recording process through a series of pre- and post-processing steps. The GUI application is implemented in Python 3.7 with Tkinter[3] and PyAudio.[4] Its interface is shown in Figure 1. Each prompt is individually displayed to the speaker. Once the recording starts, the input amplitude is monitored and its peak value is displayed such that any signal distortion or low level input can be detected. For additional monitoring, the lower panels of the interface display the waveform and spectrogram of the recorded prompt. Parameters such as sampling frequency and bit depth can be set from the configuration file and depend on the available hardware. The recording operator can easily navigate through the prompts and re-record any of them without any extra setup. Additional features of RECOApy include waveform normalization and silence trimming, as well as a *Safe Copy* option. This means that if the recording operator is unsure of the correctness of the current recording, a backup copy can be saved and later inspected.

Alongside the orthographic form of the prompts, the phonetic transcription can also be displayed. This enables the speaker to read the prompts as intended by the developer. The phonetic transcription may already be available in the prompts, or can be generated and saved from within RECOApy, as introduced in the next section.

3. G2P conversion module

To further enhance the usability and applicability of the recording tool, and given the results of [4], RECOApy can perform an accurate phonetic transcription of the prompts in eight languages: Czech, English, French, German, Italian, Polish, Romanian and Spanish. The data and methods used to develop the grapheme-to-phoneme converters are described next.

[2] https://www.audacityteam.org/
[3] https://wiki.python.org/moin/TkInter
[4] https://pypi.org/project/PyAudio/

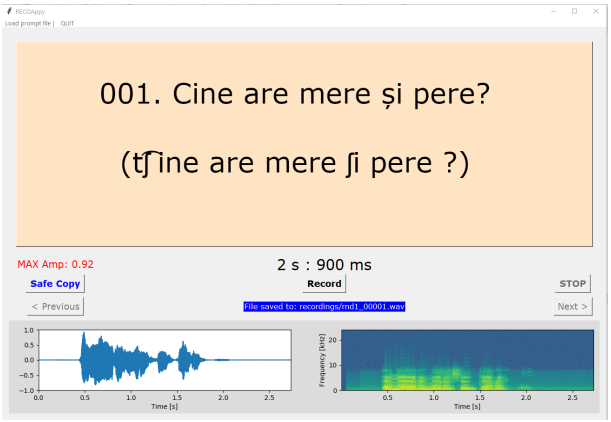

Figure 1: *RECOApy GUI*

3.1. Phonetic lexicons

Even for the mainstream languages, large, manually annotated lexicons are not easily and readily accessible. And most research groups have developed their internal resources [9, 14]. An alternative to this individual effort is the collaborative online resource called Wiktionary. It contains word definitions in 171 languages, of which 45 languages include more than 100,000 entries. The usability of Wiktionary as an alternative to the hand crafted resources has already been studied—[17] shows its great impact on the future directions of lexicography. A significant number of the dictionary entries also include phonetic transcriptions. Their use in G2P methods has been tackled before [7, 8], and can therefore constitute the base for the work presented in this paper.

However, as this resource is constantly expanding, processing the latest database dumps is beneficial [7].[5] A first step for preparing the lexicons was to determine the list of words which include phonetic entries and to extract these pronunciations. Because the data is crowd-controlled, there is no guarantee that the transcriptions are correct and consistent, or that the entries pertain to a single language. To mitigate these issues, a part of the transcriptions were discarded: entries containing graphemes outside the standard alphabet of the respective language; entries containing phonemes whose occurrence is less than 100 across the respective lexicon; and entries with a phonetic transcription significantly longer than the orthographic form, which might be indicative of two or more pronunciation versions entered in the same field. There was also a set of identical entries (same word, same phonetic transcription), and these were collapsed into a single entry. All lexical stress symbols, if present, were removed. The final number of entries in each lexicon can be found in Table 2.

Due to the potential transcription errors present in Wiktionary, which might affect the performance of the G2P conversion networks, two well-established manually checked lexicons were also included in the evaluation: the English CMU Pronunciation Dictionary[6] and the Romanian MaRePhor lexicon [18]. Version 0.7b of CMUdict was used and all entries containing numbers and any other symbol except the apostrophe were discarded. The lexical stress in the pronunciation was removed.

[5] *wiktionary-20200301** versions of the database were used here.
[6] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

3.2. G2P conversion networks

Given the variable lengths of the orthographic and phonetic representations of a word, the task of grapheme-to-phoneme conversion is inherently a sequence-to-sequence mapping problem [19]. Within the set of sequence-to-sequence deep learning algorithms, the most prominent are those based on recurrent (RNN), convolutional (CNN) and full-attention (Transformer) architectures. Although the RNN seq2seq is a highly efficient and adequate method to process temporal or order dependent sequences, it exhibits a slow convergence and high computational complexity. As a result, more and more NLP tasks have been addressed with CNN or hybrid seq2seq alternatives [20, 21]. Along the CNN-based architecture, the Transformer network has been successfully applied in machine translation tasks [22], and G2P conversion networks [9, 23].

These two seq2seq architectures were selected as the starting point in the development of RECOApy's G2P module. The CNN network's encoder and decoder are composed of 1D convolution, activation and normalization layers. An attention layer merges the hidden representations of the encoder and decoder. The attention context is concatenated with the decoder representation and passed through another set of 1D convolution layers—denoted as *decoder output*—to generate a softmax output. No residual connections or embedding layers are used. The Transformer network closely follows the architecture of [22], with multi-head self-attention layers combined with fully connected ones in the encoder, decoder and decoder output modules. A positional embedding layer pre-processes the inputs.

For these two neural architectures, the topologies which obtained the best results for English are described in [9, 23, 24]. However, taking into consideration the G2P complexity across languages, as well as the variable dimension of each phonetic lexicon, the architectures' hyperparameters need to be optimized [25]. Genetic algorithms and evolution strategies manage to provide near-optimal solutions for complex tasks, such as image classification [26] and reinforcement learning [27]. For the current task of G2P conversion across multiple languages and datasets, an evolution strategy (ES) similar to the one described in [26] was adopted. The genes represent various topology parameters, such as number of layers in the encoder or the decoder, the hidden dimensions of the layers or the activation function. The fitness of a genome is determined on its ability to predict a set of word-level phonetic transcriptions. The initial population is randomly selected from the genome pool. In each new generation, the fittest individuals are maintained and bred to create new individuals by random recombinations and mutations. A small sample of the less fit individuals are also bred in order to explore the gene space more thoroughly.

3.3. G2P results

The neural network architectures' hyperparameters were optimized over 10 generations each with a population size of 10. The fitness of a genome was assessed in terms of the word error rate (WER) computed over a held-out test set of 500 samples at the end of a 20 epoch training process. The small number of epochs and evaluation samples was chosen so that the evolution strategy did not fit the respective train-test split. The number of lexicon entries used for hyperparameter optimisation was limited to 150,000 samples.[7] The set of genes and gene values for each neural architecture is shown in Table 1. This set does by no means explore the entire hyperparameter search space, but it does address some of the key topological variables. The fittest individual for each neural architecture, language and lexicon was selected and trained further on the entire set of entries. An early stopping criterion set to monitor variations of less than 1% in the loss metric over 50 steps prevented overfitting. An 80-20 split with random sampling was employed for training and testing the networks, respectively. The split was different from the one used in the evolution strategy, and the fitness computation data was discarded.

[7] See Table 2 for the number of entries in each lexicon.

Table 1: *Set of genes and gene values used in the evolution strategy. The first column marks the gene ID within the genome.*

Gene ID		
	CNN seq2seq	
G1	encoder layers	2, 3, 4
G2	encoder layers dimension	32, 64, 128, 256
G3	decoder layers	2, 3, 4
G4	decoder layers dimension	32, 64, 128, 256
G5	decoder output layers	2, 3, 4
G6	decoder output layers dim.	32, 64, 128
G7	activation	ReLU, Linear
G8	optimizer	Adam, RMSprop
G9	batch size	32, 64, 128, 256, 512
	Transformer seq2seq	
G1	encoder layers	2, 3, 4
G2	decoder layers	2, 3, 4
G3	embedding dimension	32, 64, 128
G4	attention heads	2, 4
G5	dropout rate	0.01, 0.05, 0.1, 0.15
G6	hidden layer dimension	32, 64, 128, 256 512, 1024
G7	batch size	32, 64, 128, 256, 512

Table 2 shows the results of the G2P conversion module. It includes the total number of entries in each lexicon next to the number of unique entries and phonetic symbols. The number of phonetic symbols represent the set of symbols used in the phonetic transcriptions. For the Wiktionary lexicons these might not fully overlap with the language's phoneme set. For each neural architecture the genes of the fittest individual are also presented. The accuracy of the G2P is reported in terms of word error rate (WER) and Levenshtein distance-based phoneme error rate (PER) [28]. For entries with multiple pronunciations, the target which minimized the PER and WER was selected.

The best performing architecture varies across languages, as well as in between lexicons of the same language, but the error rate differences are not truly significant. For example, the Romanian Wiktionary lexicon is better fitted by the CNN seq2seq, while for MaRePhor, the Transformer achieves lower WER and PER. For English, both lexicons are better fitted by the Transformer. The dataset's dimension does not seem to favour any of the architectures either, even though the number of trainable parameters is largely different. For example, the MaRePhor CNN model has 173,672 trainable parameters, and the transformer has only 71,146. But by inspecting the comparable sized lexicons in Czech and Spanish, the Transformer achieves better WER and PER for Czech, yet falls short of the CNN seq2seq in Spanish. This happens despite the fact that Czech and Spanish also exhibit comparable orthographic transparency levels [6]. One conclusion that can be directly drawn from here is that there is no universal recipe to solve the G2P task, and each solution and architecture needs to be tailored to the particular language, phonetic representation, and available resources. The absolute error rates for each language presented here are comparable or lower than the ones in [7] and [14]. But

Table 2: *Lexicon descriptions, network hyperparameters and accuracy results of the grapheme-to-phoneme module . The phonetic symbols column indicates the number of distinct phonemes found in the respective lexicon. The gene IDs are listed in Table 1. Best results for each lexicon are highlighted in boldface.*

Lang	Lexicon	Entries	Unique entries	Phonetic symbols	Model	G1	G2	G3	G4	G5	G6	G7	G8	G9	WER	PER
EN	**CMUdict**	132,585	123,874	39	CNN	2	128	2	128	3	128/64/32	ReLU	RMSp	512	29.82	11.41
					Transformer	4	3	64	4	0.01	512	64	-	-	**23.16**	**8.03**
	Wiktionary	71,332	48,773	39	CNN	2	128	2	128	3	128/64/32	ReLU	RMSp	256	28.92	12.39
					Transformer	4	4	32	4	0.01	128	128	-	-	**22.50**	**8.23**
RO	**MaRePhor**	72,375	72,375	40	CNN	3	64	2	32	3	64/32/32	Lin	Adam	128	2.64	0.5
					Transformer	2	4	32	2	0.05	64	64	-	-	**2.30**	**0.42**
	Wiktionary	63,013	62,733	32	CNN	3	128	2	32	3	128/64/32	Lin	Adam	512	**3.00**	**0.50**
					Transformer	3	2	64	2	0.05	64	256	-	-	3.58	0.71
CZ	Wiktionary	42,014	41,419	41	CNN	2	32	4	128	3	64/32/32	Lin	RMSp	128	11.69	3.84
					Transformer	2	2	32	2	0.05	64	32	-	-	**9.45**	**2.37**
DE	Wiktionary	327,296	315,793	51	CNN	3	128	3	32	3	128/64/32	ReLU	Adam	512	**5.50**	**1.43**
					Transformer	4	2	64	2	0.05	32	64	-	-	8.80	2.24
ES	Wiktionary	49,346	42,732	31	CNN	3	128	4	64	2	128/64	ReLU	Adam	128	**9.81**	**2.20**
					Transformer	2	4	32	4	0.05	32	32	-	-	11.90	2.95
FR	Wiktionary	1,121,714	1,115,343	35	CNN	3	128	3	32	3	128/64/32	ReLU	Adam	512	**4.38**	1.02
					Transformer	2	3	64	2	0.05	128	64	-	-	4.78	**0.97**
IT	Wiktionary	29,826	29,242	28	CNN	2	128	4	128	2	64/32	ReLU	RMSp	256	**18.67**	**4.44**
					Transformer	2	2	64	2	0.01	512	64	-	-	19.04	5.00
PL	Wiktionary	35,646	35,544	48	CNN	4	64	2	128	2	128/64	ReLU	Adam	128	3.59	1.84
					Transformer	3	2	64	4	0.05	1024	128	-	-	**2.98**	**1.34**

the different lexicon versions and train-test splits make a direct, fully correct comparison impossible. As an overview of the architectures' performance, the average WER across lexicons for the CNN seq2seq is 11.80%, and the PER is 3.95%. For the Transformer, the average WER is 10.95% and PER is 3.22%.

Inspecting the performance over the supervised lexicons, for MaRephor the results are in line with previous studies [29]. The CMUdict error rates obtained here (23.16% WER and 8.03% PER) are slightly lower than the ones reported in the state-of-the-art methods ([23]: 22.1% WER and 5.1% PER). However, the CMUdict versions and train-test splits are different. When applying the same architecture[8] on this version of the CMUdict, the results were 22.8% WER and 7.19% PER. It is interesting, however, to notice that the ES evolved a rather similar architecture for the Transformer seq2seq. It may be the case that an evaluation of the fitness over larger number of epochs and validation set, would yield the same architecture, and therefore same performance. One other interesting fact in the results reported here is the high WER for Italian. When analysing the decoded sequences from both networks, it was found that over 60% of the erroneous words had only a single incorrect phoneme, and it was mostly the case of vowel-semivowel substitutions.

Looking at the inference duration, the MaRePhor CNN seq2seq model processes 5000 words in approximately 55 seconds, while the Transformer seq2seq does it in around 120 seconds.[9] Given the large difference in inference time and only minor drops of accuracy for some of the lexicons, RECOApy integrates the CNN-based models alone. However, the trained Transformer models are available in the tool's webpage.

[8]The authors of [23] kindly provided their implementation.

[9]On an NVidia GeForce RTX 2080 Ti GPU with 12GB vRAM.

4. Conclusions

This paper introduced RECOApy, a tool for data recording, preprocessing and phonetic transcription of training data aimed at speech-based end-to-end applications. The tool enables fast and accurate recording of text prompts at various sampling rates and bit depths, while offering the recording operator the possibility to supervise the quality of the process as well. Additional automatic options to normalise the audio and to discard the start and end silence segments are also available. One other important feature of RECOApy is that of automatic phonetic transcription of the prompts in eight languages: Czech, English, French, German, Italian, Polish, Romanian and Spanish. The G2P module consists of state-of-the-art neural network based architectures achieving low word and phoneme error rates across all languages. As a conclusion, the RECOApy tool can most certainly be used as a reliable means to develop the training data for end-to-end speech-based applications. In fact, our research group has already collected over 50 hours of prompted speech from non-expert volunteers using this recording tool. The tool, lexicons and models are available here: `www.gitlab.utcluj.ro/sadriana/recoapy/`.

Future developments of the tool include the addition of more languages in the G2P module, a more in-depth analysis of the hyperparameter space, as well as the augmentation of the prompts with syllabification and lexical stress assignment. A potential significant developement would be to also include prosodic cues—similar to [30].

5. Acknowledgement

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

6. References

[1] A. Y. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," *ArXiv*, vol. abs/1412.5567, 2014.

[2] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions," in *Proceedings of ICASSP*, 2018.

[3] S. Ö. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, J. Raiman, S. Sengupta, and M. Shoeybi, "Deep Voice: Real-time Neural Text-to-Speech," in *Proceedings of ICML*, 2017.

[4] J. Fong, J. Taylor, and K. Richmond, "A Comparison of Letters and Phones as Input to Sequence-to-Sequence Models for Speech Synthesis," in *Proc. of Interspeech*, 2019, pp. 223–227.

[5] V. Wan, C. an Chan, T. Kenter, J. Vit, and R. Clark, "Chive: Varying prosody in speech synthesis with a linguistically driven dynamic hierarchical conditional variational network," in *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*, 2019, pp. 3331–3340.

[6] G. Gillon, *Phonological Awareness 2nd Edition: From Research to Practice*. The Guilford Press, 2018.

[7] T. Schlippe, S. Ochs, and T. Schultz, "Web-based tools and methods for rapid pronunciation dictionary creation," *Speech Communication, 118, January 2014.*, vol. 56, p. 101, 2014.

[8] A. Deri and K. Knight, "Grapheme-to-phoneme models for (almost) any language," in *Proceedings of the 2016 Conference of the Association for Computational Linguistics*. Berlin, Germany: Association for Computational Linguistics, August 2016.

[9] M. Yu, H. Nguyen, A. Sokolov, J. Lepird, K. Sathyendra, S. Choudhary, A. Mouchtaris, and S. Kunzmann, "Multilingual Grapheme-to-Phoneme Conversion with Byte Representation," in *Proc. of ICASSP*, 2020.

[10] J. Route, S. Hillis, I. Czeresnia Etinger, H. Zhang, and A. W. Black, "Multimodal, Multilingual Grapheme-to-Phoneme Conversion for Low-Resource Languages," in *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 192–201.

[11] K. Yao and G. Zweig, "Sequence-to-sequence neural net models for grapheme-to-phoneme conversion," in *Proc. of Interspeech*, 2015.

[12] K. Rao, F. Peng, H. Sak, and F. Beaufays, "Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks," in *ICASSP*, 2015.

[13] B. Milde, C. Schmidt, and J. Köhler, "Multitask Sequence-to-Sequence Models for Grapheme-to-Phoneme Conversion," in *Proc. of Interspeech*, 2017.

[14] D. van Esch, M. Chua, and K. Rao, "Predicting pronunciations with syllabification and stress with recurrent neural networks," in *Proceedings of Interspeech*, 2016.

[15] M. Juzová, D. Tihelka, and J. Vit, "Unified Language-Independent DNN-Based G2P Converter," in *Proceedings of Interspeech*, 2019.

[16] K. Sjölander and J. Beskow, "Wavesurfer - an open source speech tool." in *Proceedings of Interspeech*. ISCA, 2000, pp. 464–467.

[17] C. M. Meyer and I. Gurevych, "Wiktionary: A new rival for expert-built lexicons? exploring the possibilities of collaborative lexicography," in *Electronic Lexicography*, S. Granger and M. Paquot, Eds. Oxford: Oxford University Press, November 2012, pp. 259–291.

[18] 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.

[19] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14, 2014, p. 3104–3112.

[20] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional Sequence to Sequence Learning," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70, 2017, pp. 1243–1252.

[21] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of CNN and RNN for natural language processing," *CoRR*, vol. abs/1702.01923, 2017.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008.

[23] S. Yolchuyeva, G. Németh, and B. Gyires-Tóth, "Transformer based grapheme-to-phoneme conversion," *Proceedings of Interspeech*, Sep 2019.

[24] ——, "Grapheme-to-Phoneme Conversion with Convolutional Neural Networks," *Applied Sciences*, vol. 9, no. 6, p. 1143, 2019.

[25] G. Melis, C. Dyer, and P. Blunsom, "On the state of the art of evaluation in neural language models," *CoRR*, vol. abs/1707.05589, 2017.

[26] T. Hinz, N. Navarro, S. Magg, and S. Wermter, "Speeding up the hyperparameter optimization of deep convolutional neural networks," *International Journal of Computational Intelligence and Applications*, vol. 17, pp. 1 850 008:1–1 850 008:15, 2018.

[27] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning," *CoRR*, vol. abs/1712.06567, 2017.

[28] V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, 1966.

[29] A. Stan, "Input Encoding for Sequence-to-Sequence Learning of Romanian Grapheme-to-Phoneme Conversion," in *Proceedings of the 10th IEEE International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, Timisoara, Romania, October, 10-12 2019.

[30] R. Wilhelms-Tricarico, J. B. Reichenbach, and G. Marple, "The Lessac Technologies Hybrid Concatenated System for Blizzard Challenge 2013," in *Proceedings of Blizzard Challenge*, 2013.