Multitask Sequence-to-Sequence Models for Grapheme-to-Phoneme Conversion

Benjamin Milde, Christoph Schmidt, Joachim Köhler

Fraunhofer Institute IAIS, Sankt Augustin, Germany

Benjamin.Milde@, Christoph.Andreas.Schmidt@, Joachim.Koehler@iais.fraunhofer.de

Abstract

Recently, neural sequence-to-sequence (Seq2Seq) models have been applied to the problem of grapheme-to-phoneme (G2P) conversion. These models offer a straightforward way of modeling the conversion by jointly learning the alignment and translation of input to output tokens in an end-to-end fashion. However, until now this approach did not show improved error rates on its own compared to traditional joint-sequence based n-gram models for G2P. In this paper, we investigate how multitask learning can improve the performance of Seq2Seq G2P models. A single Seq2Seq model is trained on multiple phoneme lexicon datasets containing multiple languages and phonetic alphabets. Although multi-language learning does not show improved error rates, combining standard datasets and crawled data with different phonetic alphabets of the same language shows promising error reductions on English and German Seq2Seq G2P conversion. Finally, combining Seq2seq G2P models with standard n-grams based models yields significant improvements over using either model alone.

Index Terms: grapheme to phoneme, G2P, sequence to sequence, Seq2Seq , multitask learning

1. Introduction

A crucial component of most automatic speech recognition (ASR) systems is the phoneme lexicon, mapping words to their phonetic representation (e.g. Thursday $\rightarrow$ TH ER Z D EY). Creating and maintaining phoneme lexicons manually is a tedious task and needs expert phoneticians. Typically, a seed lexicon is used to train a model that can automatically produce phonetic entries, to either aid expert phoneticians or (accepting a certain error rate) to generate new entries fully automatically.

Recently, neural sequence-to-sequence models (Seq2Seq) have emerged as a generic approach to learn to translate between sequences of varying lengths [1]. Initially applied to machine translation, Seq2Seq has been successfully applied to a wide range of various other tasks, including conversation modelling [2], sentence-level grammatical error identification [3], automatic regex generation [4] and also recently grapheme-to-phoneme (G2P) conversion [5]. Seq2Seq models are generative language models, conditioned on an input sequence. After encoding the input sequence token by token, the output sequence is generated token by token. No explicit alignments between input and output sequences are necessary, as the system is trained in an end-to-end fashion.

We mainly target the use of the G2P model in automatic speech recognition. Training and using a G2P model is often directly integrated into the ASR training procedure, as phonetic out-of-vacabulary (OOV) words in the training set hamper the alignment of training data to its transcriptions. Common words as determined by the language model that do not have a pronunciation entry can also be candidates for G2P conversion and can be used to extend the vocabulary of a large vocabulary speech recognition system. Other uses for G2P conversion include generating OOV entries on the fly for text to speech systems and assisting humans in the generation of phoneme lexicons.

The next section covers related work on grapheme-to-phoneme conversion and sequence-to-sequence models. We give an overview over neural sequence-to-sequence models in Section 3, describe our evaluation in Section 4, compare Seq2Seq models with and without multitask learning in Section 5 and finally conclude in Section 6.

2. Related Work

Joint-sequence n-gram models are both well performing and popular traditional models for grapheme-to-phoneme conversion [6, 7]. These models need to find a joint vocabulary of graphemes and phonemes (often called graphones) by aligning characters and phonemes. The output sequence is modelled as a sequence of graphones. Particularly Sequitur G2P [7] is a well established G2P conversion tool using joint sequence modelling and is also commonly used in state-of-the-art speech recognition model recipes, e.g. [8]. Joint n-gram models can also be efficiently represented as weighted finite state transducers [9, 10].

Yao and Zweig described the first application of Seq2Seq to English G2P in [5]. The decoder/encoder network did yield 17% higher WERs than Sequitur G2P. However, they achieved state of the art results on three G2P datasets by using a fixed length recurrent architecture together with the HMM many-to-many alignment procedure from [11]. Rao et al. [12] proposes to use the connectist temporal classification (CTC) instead of Seq2Seq to jointly align and translate graphemes to phonemes in a single neural model. The proposed model on its own yields 5% higher word error rates than the Sequitur G2P baseline. However, when both models are combined with a finite state transducer (FST) n-gram model, they significantly outperform the baseline. Schnober et al. also reported a negative result for sequence-to-sequence models with an attention mechanism on a G2P task with 20k lexicon entries (among other monotone string translation tasks), compared to standard methods [13].

In [14], Tsvetkov et al. trained phonetic language models in a multitask training setting. Perplexities of the joint model over all languages are lower than individual phonetic language models. State of the art models for translation use neural sequence-to-sequence models and incorporate an attention mechanism and residual learning [15, 16]. This model can also be trained on multiple language pairs at once: an additional unique language identifier token at the beginning of the input sentence is added to specify the required target language to translate to [17]. We follow this multitask learning [18] approach to train multi-

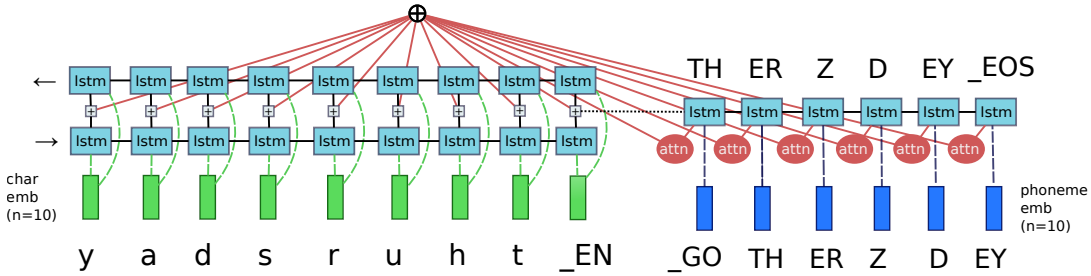

Figure 1: *Seq2Seq model for G2P conversion with attention and character/phoneme embeddings, inputs are reversed. For multitask learning, we extend the source vocabulary with additional markers for the subtask that are placed at the beginning of each word.*

language and multi-alphabet grapheme-to-phoneme conversion models.

3. Neural Sequence-to-Sequence Models

Neural sequence-to-sequence models can learn a conditional distribution over a variable length sequence conditioned on another sequence $p(y_1, ..., y_T|x_1, ..., x_{T'})$, where T' can be different from T [19]. When doing phoneme-to-grapheme conversion, we condition the output phoneme sequence $y_1, ..., y_T$ on the character sequence $x_1, ..., x_{T'}$. In the following, we are describing the Seq2Seq that we are going to use for our G2P experiments.

Plain Seq2Seq models. A plain Seq2Seq model follows an encoder and decoder design, where the input sequence is encoded token by token and the output sequence is generated token by token. Typically, recurrent neural networks are used in both the decoder and encoder network. The RNN unit is usually one with a gating mechanism, e.g. a Long-Short Term Memory (LSTM) [20] or a Gated Recurrent Unit (GRU) [21], to counter the vanishing gradient problem inherent in RNNs [22]. The output sequence $y_1, ..., y_T$ is conditioned on a single vector that is generated with the encoder of the network, by initializing the RNN decoder state to the last RNN state of the encoder.

Encoder/decoder inputs. We use character embeddings for the encoder and phoneme embeddings for the decoder as inputs. For both we choose n=10 as the embedding size. The embeddings are trained as part of the model training. Following [1], we reverse the input character sequence.

Attention mechanism. A drawback of the plain Seq2Seq model is that all the source information has to pass through a single vector and needs to be carried forward in the decoder's state. In order to facilitate information flow from the source sequence to the target sequence and to relieve the decoder from having to remember the input sequence in its state, a context vector can be computed as a weighted sum over all encoder hidden states, dependent on the last decoder state [23]. The attention vector is either added or appended to the RNN decoder inputs. Different variations of this mechanism exist, which can be divided into local and global attention mechanisms [24]. We use global attention, where an attention vector is generated at each phoneme generation step over the full character input sequence.

Stacked bidirectional LSTM. The encoder can also be extended to represent past and future dependencies, with two RNNs that read a word in opposite directions. Figure 1 depicts a Seq2Seq architecture for G2P with a bidirectional LSTM encoder using a decoder with global attention. Furthermore, LSTM cells can also be stacked vertically, by passing the output of a LSTM cell as input to another LSTM cell. We combine the outputs of the forward and backward encoder with a vector sum. (An alternative would be to stack both vectors).

Residual connections. The residual network [15] provides skip-connections between layers of a network. The main idea is that learning the identity function is simpler with a residual connection, because bypassing the layers computation means that the output of a layer needs to produce a zero output and not the identity function. We make use of a simple residual connection between LSTM layers, where the output of an LSTM layer is added to its input before passing it to the next LSTM layer:

$$y = F(x; W) + x$$

where y is the output of a layer and x the input. $F(x; W)$ is a function with an internal parameter W [25]. We use the LSTM unit directly as F when stacking LSTM units, i.e. we do not add skip connections between timesteps.

Decoding. At each decoding step, the decoder generates a softmax distribution over the output vocabulary. The simplest method to generate the output sequence is to feed the argmax token to the next decoding step. However, a greedy decoder might not be able to find the sequence with the best joint probability. Beam decoding can also be used to keep n different paths while decoding, while still efficiently searching through the search space.

4. Evaluation

We base our evaluation on German and English G2P conversion, by comparing the model predictions on unseen data to manual entries by expert phoneticians. For German, we use Phonolex[1] core (66.8k entries) and the full Phonolex lexicon (1.4 million entries), that uses the SAMPA phonemes set. All entries in the core set are manually verified entries, while the source of the other entries is sometimes unclear, with most pronunciations entries of the full set likely to be already coming from rule based conversion systems [26], automatic G2P conversions and data-driven alignments, e.g. BAS Maus [27]. For English we use CMUDICT 0.7b with about 138k manual pronunciation entries using the ARPAbet phoneme set. For all training scenarios, we preprocess the lexicon to remove stress markers, which are usually omitted in ASR acoustic modelling.

We evaluate phoneme error rate (PER) and phoneme word error rate (WER) by splitting the available data into training, development and test splits. For Phonolex (core/full), we only use manually verified entries as development and test data, i.e.

[1] `http://www.phonetik.uni-muenchen.de/Bas/BasPHONOLEXdeu.html`

Table 1: *Comparing Sequitur G2P and seq2seq-attn on the German Phonolex lexicon test data, with stress markers removed. (Best scores for single models and system combinations in bold.)*

Model	Phonolex set	PER	WER	train time
(1) Sequitur G2P (model order 10)	core set	**1.98%**	11.54%	12h43
(2) Sequitur G2P (model order 6)	core set	**1.98%**	**11.30%**	3h40
(3) Sequitur G2P (model order 6)	full set	5.41%	29.86%	7.45 days
(4) seq2seq-attn (biLSTM 256x3, d=0.5)	full set	6.09%	32.69%	13h57
(5) seq2seq-attn (biLSTM 256x3, d=0.5)	core set	2.49%	13.64%	3h59
(6) seq2seq-attn (biLSTM residual 256x3, d=0.5)	core set	2.37%	12.75%	3h53
(7) seq2seq-attn (biLSTM residual 256x3, d=0.5) + multitask learning (de/en)	core set	2.57%	14.12%	5h51
(8) seq2seq-attn (biLSTM 512x3, d=0.5) + multitask learning (de/en)	core set	2.41%	13.32%	8h27
(9) seq2seq-attn (biLSTM res. 512x3, d=0.5) + multitask learning (Sampa/IPA)	core set	2.06%	**11.30%**	24h49
(10) System combination (2)+(6)	core set	1.88%	10.33%	(7h39)
(11) System combination (2)+(9)	core set	**1.70%**	**9.52%**	(28h29)

Table 2: *WER and PER performance for different classes of words in the German Phonolex task. While regular German words can be phonetized with relatively small errors, loan words and named entities are particularly problematic in this G2P task.*

	Abbreviation		English pronoun.		French pronoun.		Name / NE		Hyphen		Regular	
Count and % of all entries	23 (1.75%)		34 (2.59%)		8 (0.61%)		149 (11.36%)		29 (2.21%)		996 (75.91%)	
Model	**PER**	**WER**	**PER**	**WER**	**PER**	**WER**	**PER**	**WER**	**PER**	**WER**	**PER**	**WER**
Sequitur G2P (model order 6)	**2.92%**	**8.70%**	**10.20%**	**38.24%**	35.00%	**62.50%**	**5.00%**	**23.49%**	2.43%	31.03%	1.13%	7.63%
seq2seq-attn (biLSTM 256x3)	10.59%	26.09%	16.93%	52.94%	32.81%	75.00%	5.80%	25.50%	**1.22%**	**13.79%**	1.18%	8.63%
+ multitask (de/en)	8.40%	26.09%	17.20%	61.76%	25.05%	**62.50%**	5.41%	26.10%	**1.22%**	17.24%	1.39%	9.24%
+ multitask (SAMPA/IPA)	6.72%	21.74%	12.60%	47.06%	**22.22%**	**62.50%**	5.58%	24.16%	2.22%	17.24%	**1.07%**	**7.33%**

we use 1312 entries (2%) for each set of Phonolex core. For CMUDICT, we use the same train/dev/test splitting as proposed in the sequence-to-sequence example of the CNTK toolkit[2], i.e. 108952 train entries, 5447 dev entries and 12855 test entries. While Phonolex core does not contain pronunciation variants, CMUDict does include them, and we also use them in the training process. Following Bisani and Ney [7], the variant that minimizes the error is chosen for computing PER and WER.

To enable multi-alphabet training, we also generated our own German and English dictionaries using the International Phonetic Alphabet (IPA)[3]. We used recent dumps of the German and English Wiktionary project[4], containing IPA phoneme entries of voluntary contributers (not all lexicon entries have a phoneme entry). For German, we pick the top pronunciation entry. For English we include all variants, but not British ones (to favor American English, which CMUDICT also models). Otherwise, no attempt at normalizing notation styles is made. The English Wiktionary contains 4 million entries, but the biggest fraction are non-English words, mostly with the exact pronunciation of the original language. Accordingly, by only using IPA entries that are clearly marked as German or English entries, we obtain an IPA lexicon of 330.473 words for German and a significantly smaller IPA lexicon of 34.943 words for English.

For Sequitur G2P, we tune the model order (n-gram) on the development set. While 6 is the recommended parameter of the training software, we found higher order models to give slightly better results on the development set. It is also used in the training procedure to adjust the discount parameters of the joint grapheme/phoneme language model.

For our sequence-to-sequence experiments we use seq2seq-attn[5], an open source neural translation system. It can be fitted to G2P conversion by preprocessing the dataset so that single characters or phonemes are entire words. We also tune Seq2Seq models on the development set: we choose the width of the networks' LSTM layers from the set {64, 128, 256, 512}, number of layers from {2, 3, 4, 5} and dropout from {0.0, 0.2, 0.3, 0.5} and train with and without residual learning. We run all 128 configurations on a cluster with Nvidia Titan X GPUs and select promising candidate configurations on the development set. We omit hyperparameter optimization entirely for the full Phonolex set and leave out the 64 and 128 width configurations for multi-task training. We exclusively use Adam [28] as optimization method for network training, with an initial learning rate of 0.001. For the first 3 epochs we use curriculum learning (sorting the sequences by length in an epoch), for all other epochs the order is randomized.

In order to enable system combinations, we also let Sequitur G2P and seq2seq-attn generate n-best lists (n=10). We simply combine the n-best lists by adding normalized scores between 0.0 and 1.0 for entries that both models generated and append all other entries. After resorting, the output of the system combination is the highest scoring entry from the combined n-best list.

[2] https://github.com/Microsoft/CNTK/tree/master/Examples/SequenceToSequence/CMUDict/Data

[3] Scripts to reproduce the IPA dictionary generation are available at https://github.com/bmilde/wiktionary_ipa_phoneme_lexicons

[4] dewiktionary-20170120 and enwiktionary-20170301

[5] https://github.com/harvardnlp/seq2seq-attn

Table 3: *Comparing Sequitur G2P and seq2seq-attn on a CMUDICT test set, with stress markers removed. Results shown here are for a recent version of the lexicon (v0.7b).*

Model	PER	WER	train time
(1) Sequitur G2P (model order 8)	**6.12%**	**25.71%**	34h49
(2) seq2seq-attn (biLSTM 256x4, d=0.3)	6.81%	29.51%	9h38
(3) seq2seq-attn (biLSTM 256x4 residual, d=0.3)	6.67%	28.72%	9h40
(4) seq2seq-attn (biLSTM 512x3) + multitask learning (en/de)	6.68%	29.28%	8h27
(5) seq2seq-attn (biLSTM 512x3) + multitask learning (ARPAbet/IPA)	6.5%	28.23%	6h30
(6) System combination (1) + (3)	5.91%	25.15%	(44h29)
(7) System combination (1) + (5)	**5.76%**	**24.88%**	(41h19)

5. Results

In Table 1, we compare different G2P models on the Phonolex lexicon test set. Using the full lexicon as opposed to Phonolex core to train the models (the test set is the same in both cases) significantly reduces performances (3, 4). We attribute this to the problem that many entries of the full Phonolex set are already automatically generated in some way. We were able to spot several problematic non-core entries, mostly in entries marked as coming from automatic (and probably old) rule-based conversions. We thus focused on training with the core set after a few experiments. Seq2Seq models (5) and (6) do perform worse as a single model than the Sequitur G2P baselines (1) and (2). Training a multitask model on Phonolex and CMUDICT training data (7, 8) did not improve upon the residual network baseline (6), but is slightly better than a Seq2Seq model without residual learning (5). However, if we train a multitask model on our generated German IPA lexicon and Phonolex core (9), we can considerably lower error rates, compared to all other Seq2Seq models. With this model, we can also match the WER score of the Sequitur G2P baseline model.

We observed that the multitask models generally need wider networks, to account for the increased need of model capacity to model two G2P tasks. Residual learning can give a small performance improvement as in (5) vs. (6), and also was useful for (7) and (9), as tested on the dev set. Finally, if we combine the Sequitur G2P model (2) with our Seq2Seq models using n-best lists, we achieve the lowest error rates (10, 11).

To better understand the scores of the single models on the German Phonolex data, we annotated the Phonolex test set into the following irregular pronunciation cases: abbreviations (e.g. "GPA", "FFH", "MIDI-Dateien"), loan words and names with predominately English pronunciation (e.g. "Toaster", "mousepad"), the same for French (e.g. "arrangierter", "Taille"), other names and named entities (e.g. "Dietmar", "Chemnitz") as well as hyphenated words (e.g. "Acht-Uhr-fünf-Flug"). The irregular pronunciation cases make up 24.09% of the data. We also discarded 73 (5.56%) of the test entries[6] from Phonolex, containing spelling errors of words, unknown words (e.g. "Beteginsdis") and transliterations of Bavarian dialects (e.g. "obgnomma", "ausaschauen" for standard German "abgenommen", "hinausschauen").

Table 2 shows PER and WER results separately for our word classes. As expected, there is a large increase in error rates for irregularly pronounced German words and other special forms across all models. Interestingly, Sequitur G2P struggles less than the Seq2Seq models with abbreviations, English and French loanwords and names[7]. Hyphenated words are considerably more problematic for Sequitur G2P than for Seq2Seq models. The multitask model trained on Sampa/IPA data mainly improves regular words compared to Sequitur G2P. Although still showing a bit higher error rates on irregular cases than Sequitur G2P, it shows lower error rates for most classes of irregular words compared to the other Seq2Seq models.

In Table 3, we also compare similar models on English G2P conversion, as a control experiment. Similarly, Sequitur G2P (1) outperforms individual Seq2Seq models. Residual learning gives a small improvement (3) vs. (2) and the multi-task model trained on German and English data (4) does degrade error rates very slightly, similarly to the German test scores in Table 1. The multitask model for English ARPAbet and IPA entries does slightly improve the Seq2Seq scores (5), but fails to match the Sequitur G2P error rate on the CMUDICT test set (1). This can be attributed to the much smaller size of the auxiliary IPA lexicon, as it is 10 times smaller than in the German experiment and also smaller than CMUDICT itself. Again, if we use a simple combination of the model's n-best lists (6, 7), we observe lower error rates than from any individual model.

6. Conclusion

Our results still largely favor traditional joint sequence n-gram modelling (with Sequitur G2P) if individual models are compared, similarly to related work. However, any further advances in either RNN architectures or the Seq2Seq architecture will likely also benefit Seq2Seq G2P. Simultaneous training on the German and English G2P task did no yield any benefits, but our multitask Sampa/IPA Seq2Seq model for German G2P performs on par with the Sequitur G2P baseline and shows significantly lower error rates than the other Seq2Seq models.

Our error analysis according to word classes shows that there is diversity in how the models learn to deal with regularly and irregularly pronounced German words, which are difficult to predict. Ultimately, this diversity allows us to combine different systems: combining the Sequitur G2P and Seq2Seq models without multitask learning (8.6% lower WER relative) and with multitask learning (15.8% lower WER relative) yield significant WER improvements for German G2P.

The results are similar for English G2P on CMUDICT, but since the auxiliary IPA lexicon is 10 times smaller the German one, we can only observe a modest decrease in WER for multitask Seq2Seq models. We still think that multitask learning is a simple way to combine non-heterogeneous lexicons in Seq2Seq G2P models, without the need for an explicit translation between different phoneme sets and annotation styles.

Acknowledgments. We thank Gunnar Åkermark and Michael Stadtschnitzer for many helpful comments, suggestions and discussions.

[6]Discarded words are also discarded from the analysis in Table 1.

[7]The results for French loan words have to be taken with grain of salt though, as the sample size is very small (n=8).

7. References

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Proc. NIPS*, Montreal, Canada, 2014, pp. 3104–3112.

[2] O. Vinyals and Q. Le, "A Neural Conversation Model," in *Proc. of ICML Deep Learning Workshop*, Lille, France, 2015.

[3] A. Schmaltz, Y. Kim, A. M. Rush, and S. M. Shieber, "Sentence-level Grammatical Error Identification as Sequence-to-sequence Correction," in *NAACL HLT*, San Diego, CA, USA, 2016, pp. 242–251.

[4] N. Locascio, K. Narasimhan, E. DeLeon, N. Kushman, and R. Barzilay, "Neural Generation of Regular Expressions from Natural Language with Minimal Domain Knowledge," *arXiv preprint arXiv:1608.03000*, 2016.

[5] K. Yao and G. Zweig, "Sequence-to-sequence Neural Net Models for Grapheme-to-phoneme Conversion," *arXiv preprint arXiv:1506.00196*, 2015.

[6] L. Galescu and J. F. Allen, "Bi-directional Conversion Between Graphemes and Phonemes using a Joint N-gram Model," in *4th ISCA Tutorial and Research Workshop (ITRW) on Speech Synthesis*, Perthshire, Scotland, 2001.

[7] M. Bisani and H. Ney, "Joint-sequence Models for Grapheme-to-phoneme Conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.

[8] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR Corpus Based On Public Domain Audio Books," in *Proc. ICASSP*, Brisbane, Australia, 2015, pp. 5206–5210.

[9] J. R. Novak, N. Minematsu, and K. Hirose, "Failure Transitions for Joint N-gram Models and G2P Conversion." in *Proc. Interspeech*, Lyon, France, 2013, pp. 1821–1825.

[10] J. R. Novak, "Phonetisaurus," 2012. [Online]. Available: https://github.com/AdolfVonKleist/Phonetisaurus

[11] S. Jiampojamarn, G. Kondrak, and T. Sherif, "Applying Many-to-Many Alignments and Hidden Markov Models to Letter-to-Phoneme Conversion," in *Proc. NAACL-HLT*, Rochester, New York, 2007, pp. 372–379.

[12] K. Rao, F. Peng, H. Sak, and F. Beaufays, "Grapheme-to-phoneme Conversion using Long Short-term Memory Recurrent Neural Networks," in *Proc. ICASSP*, Brisbane, Australia, 2015, pp. 4225–4229.

[13] C. Schnober, S. Eger, E.-L. D. Dinh, and I. Gurevych, "Still not there? Comparing Traditional Sequence-to-Sequence Models to Encoder-Decoder Neural Networks on Monotone String Translation Tasks," in *Proc. COLING*, Osaka, Japan, 2016, pp. 1703–1714.

[14] Y. Tsvetkov, S. Sitaram, M. Faruqui, G. Lample, P. Littell, D. Mortensen, A. W. Black, L. Levin, and C. Dyer, "Polyglot Neural Language Models: A Case Study in Cross-Lingual Phonetic Representation Learning," in *Proc. NAACL*, San Diego, CA, USA, 2016, pp. 1357–1366.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proc. CVPR*, Las Vegas, NV, USA, 2016, pp. 770–778.

[16] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, and Others, "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," *arXiv preprint arXiv:1609.08144*, 2016.

[17] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, and Others, "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation," *arXiv preprint arXiv:1611.04558*, 2016.

[18] R. Caruana, "Multitask Learning," in *Learning to learn*. Springer, 1998, pp. 95–133.

[19] K. Cho, B. van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Proc. EMNLP*, Doha, Qatar, 2014, pp. 1724–1734.

[20] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," in *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8)*, Doha, Qatar, 2014.

[22] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen Netzen," *Master's thesis, Institut fur Informatik, Technische Universitat, Munchen*, 1991.

[23] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *Proc. ICLR*, San Diego, CA, USA, 2015.

[24] T. Luong, H. Pham, and D. C. Manning, "Effective Approaches to Attention-based Neural Machine Translation," in *Proc. EMNLP*, Lisbon, Portugal, 2015, pp. 1412–1421.

[25] J. Kim, M. El-Khamy, and J. Lee, "Residual LSTM: Design of a Deep Recurrent Architecture for Distant Speech Recognition," *arXiv preprint arXiv:1701.03360*, 2017.

[26] F. Schiel, "The Bavarian Archive for Speech Signals: Resources for the Speech Community," in *Proc. Eurospeech*, Rhodes, Greece, 1997, pp. 1687–1690.

[27] N. Beringer and F. Schiel, "The Quality of Multilingual Automatic Segmentation using German MAUS," in *Proc. Interspeech 2000*, Beijing, China, 2000, pp. 728–731.

[28] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980