INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Character-Aware Sub-word Level Language Modeling for Uyghur and Turkish ASR

Chang Liu[12], *Zhen Zhang*[3], *Pengyuan Zhang*[12], *Yonghong Yan*[124]

[1]Institute of Acoustics, Chinese Academy of Sciences, China
[2]University of Chinese Academy of Sciences, China
[3]National Computer Network Emergency Response Technical Team/Coordination Center of China
[4]Xinjiang Technical Institute of Physics and Chemistry, Chinese Academy of Sciences, China
{liuchang,zhangpengyuan,yanyonghong}@hccl.ioa.ac.cn, zhangzhen@cert.org.cn

Abstract

Uyghur and Turkish are two typical agglutinative languages, which suffer heavily from the data sparsity problem. Due to this, we first apply a statistical morphological segmentation and change the number of morphs to get a better sub-word level automatic speech recognition (ASR) system. The best systems, which yield 2.03% and 1.65% absolute WER reductions from the word level systems for Uyghur and Turkish respectively, are used for further n-best rescoring. To further alleviate the data sparsity problem, we use both convolutional neural network (CNN) based and bi-directional long short-term memory (BLSTM) based character-aware language models on the two languages. In order to alleviate the information missing of the middle steps of the BLSTM based character aware language model, we propose to use the weighted average of each time-steps' outputs. The proposed weighting methods can be divided into three categories: decay based, position-based and attention-based. Results show that the decay based weighting method leads to the most significant WER reductions, which are 2.38% and 1.96%, compared with the sub-word level 1-pass ASR system for Uyghur and Turkish respectively.

Index Terms: Automatic speech recognition, sub-word language modeling, bi-directional long short-term memory network

1. Introduction

Turkish and Uyghur are two typical agglutinative languages with rich morphology, which frequently use affixes, especially suffixes. Most affixes in these two languages indicate the grammatical function of the word. New word forms can be derived from a single stem by concatenating different suffix sequences. Due to this word forming methods, one Uyghur or Turkish word may correspond to a group of English words in meaning. For example, from the Turkish word "ev", which means "house", many words can be derived by adding suffixes, like "evler" (houses), "evin" (your house), "evim" (my house), "evimde" (at my house), "evlerinizden" (from your house), "Evinizdeyim." (I am at your house.) and "Evinizdeymişim." (I was apparently at your house.). Consequently, for document with a fixed number of tokens, the number of types is usually higher for agglutinative languages than others. And the Uyghur and Turkish automatic speech recognition (ASR) systems may encounter a heavier data sparsity problem and are easier to meet a higher out-of-vocabulary (OOV) rate for a fixed number of vocabulary.

Dividing words into smaller segments, which are often called morphs, is a conventional and effective way of overcome the two problems introduced before[1, 2, 3, 4]. This is because the number of the total modeling units and the infrequent modeling units are both much smaller for morphs level systems. Among the segmentation methods, the algorithm based on minimum description length (MDL) is widely used as it does not require any linguistic knowledge. It tries to minimize the sum of text encoding length and the sub-word dictionary encoding length. By changing the ratio of the two parts in the loss of MDL, we can get segmentation models with different splitting granularities. In our previous work, we found that the consistency between language modeling units and ASR system units is important for the effectiveness of rescoring. Thus, it is necessary to choose the modeling unit of the ASR system carefully.

Another characteristic of both Uyghur and Turkish is that their alphabets represent their pronunciations with a high degree of accuracy and specificity. Specifically, each character basically corresponds to one phoneme for the two languages. This feature makes it possible to get a lexicon of the segmentation units automatically.

As its ability to model longer span dependencies, recurrent neural network language models (RNNLMs) [5, 6, 7] have been shown to be effective in ASR tasks in recent years. Many works have been done to embed character level information in word level language models. Piotr *et al.*[8] combined two networks, one working with characters at the input, and the other with words. Wang *et al.*[9] proposed to use the bi-directional long short-term memory (BLSTM) networks[10, 11] to get the word representations, which are feed to the hidden layers of the RNN language models. Later, Kim *et al.* [12] proposed to use a character level convolutional neural network (CNN) and a highway network to replace the BLSTM. The output of character-level BLSTM and CNN are used as sub-word features for the corresponding word. However, the conventional BLSTM based character-aware language model only makes use of the outputs of the last time-steps for both directions of BLSTM. As the last outputs attach more importance to the last few characters, the final output of the BLSTM could miss some important information of the middle time steps.

To deal with the problem mentioned above, we propose to use the weighed sum of the outputs of BLSTM to augment the influence of middle time-steps. The first attempt is an imitation to the conventional one by use a decay factor to weight the outputs before the last one. Beyond that, two position-based methods which use a learned vector or matrix to get the weightings are proposed. Finally we propose an attention-based method, which decides the weightings for the outputs of the BLSTM. This attention-based method is similar with the work of Zhou *et al.*[13], which is applied to relation classification task.

In this paper, we first apply the statistical morphological

http://dx.doi.org/10.21437/Interspeech.2019-1484

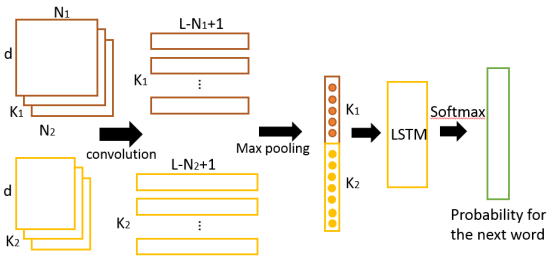

Figure 1: *The structure of CNN based character-aware language model*

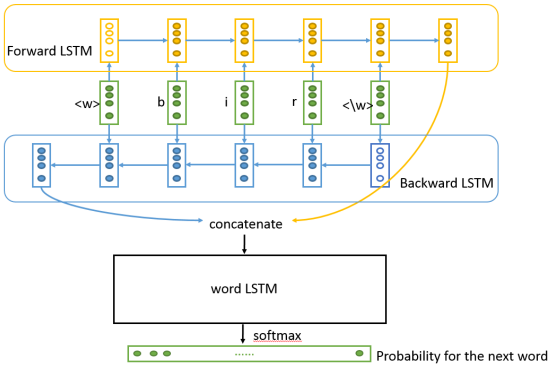

Figure 2: *The structure of BLSTM based character-aware language model*

segmentation and choose the best number of morphs by the word error rate (WER) of the corresponding ASR systems for the two languages. Then we present results of the conventional CNN and BLSTM based sub-word level language models together with the variations proposed by us on the chosen segmentation units.

2. Language Models with Sub-word Information in ASR

In this section, we introduce two widely used methods to capture sub-word features and map character level input to word embeddings.

2.1. Convolutional Neural Network on Sub-word Inputs

In this section, a character level CNN is added before the hidden layers to augment RNNLMs with sub-word features. Suppose the input word w_t is made up of a sequence of characters $(c_1, c_2, ..., c_L)$, where L is the length of the word. The input of CNN is a matrix of character level embeddings with zero-paddings to make the matrix of a certain column number. Let $M_t = [E_{c_1}, E_{c_2}, ..., E_{c_L}, 0, ..., 0] \in \mathbb{R}^{d \times l_c}$ be the padded character embedding sequence, where l_c is the length of the longest word and d is the length of the character embedding. Filters of different sizes are used to get features of character sub-sequences of different size after max pooling [12]. An illustration of CNN based neural language model is in Figure 1.

2.2. BLSTM-based Character-Aware Word Embeddings

Another structure to be added before the hidden layer to capture sub-word level features is BLSTM. BLSTM includes a forward LSTM and a backward LSTM. For the word w_t, the forward LSTM receives the sequence $c_1, c_2, ..., c_L$ as input, while the backward LSTM receives the inverse sequence, $(c_L, c_{L-1}, ..., c_1)$, as input. An instruction of the BLSTM based character aware language model is shown in Figure 2. Usually, the outputs of the final time-step of the forward and backward LSTMs are concatenated together and fed to the next hidden layer.

2.3. N-best Rescoring

In ASR tasks, RNNLMs are usually used to re-rank the n-best hypotheses from the first pass decoding due to their long history dependencies. For an utterance u_i, a list of N hypotheses $[H_1, H2, ..., H_N]$ and the corresponding acoustic model score s_{ac} and language model score s_{lm} for each hypothesis can be obtained from the first pass decoding process. The new language model score s_{nlm} of a hypothesis $H_i = (w_1, w_2, ..., w_L)$, which is of length L, is the logarithm of the probability of the sentence H_i:

$$s_{nlm} = log(P(H_i)) = log(\prod_{i=1}^{L+1} P(w_i|w_{0..i-1}) \quad (1)$$

$$= \sum_{i=1}^{L+1} log(P(w_i|w_{0..i-1})) \quad (2)$$

where w_0 and w_{L+1} are the sentence boundary sign <SOS> and <EOS>. The final language model score is linearly interpolated by the old and new language model score. And the final score of H_i is computed by

$$s = s_{ac} + (1 - \beta)s_{lm} + \beta s_{nlm}, \quad (3)$$

where β is the interpolation coefficient of the new language model.

3. Methods to Enhance the Effect of Middle time-steps

3.1. decay based methods

Let $H_{f,t} = [h_{f,1}, h_{f,2}, ..., h_{f,L}, 0, ..., 0] \in \mathbb{R}^{k \times l_c}$ and $H_{b,t} = [h_{b,1}, h_{b,2}, ..., h_{b,L}, 0, ..., 0] \in \mathbb{R}^{n \times l_c}$ be the output of the forward LSTM and backward LSTM respectively. Here, we mask the output of the zero inputs with zero. As introduced in Section 2.2, the concatenation of $h_{f,L} \in \mathbb{R}^n$ and $h_{b,1} \in \mathbb{R}^n$is the final output of the BLSTM.

To emphasize the output of former time-steps of BLSTM, we use a weighted sum of the former time-steps' output. In this subsection, we use exponential sequences to be the weighting. Let γ be the decay factor, the final outputs of the forward LSTM and the backward LSTM is calculated by the following equations respectively:

$$h_f = \frac{h_{f,L} + \sum_{i=1}^{L-1} \gamma^i h_{f,L-i}}{Z}, \quad (4)$$

$$h_b = \frac{h_{b,1} + \sum_{i=1}^{L-1} \gamma^i h_{f,i+1}}{Z}, \quad (5)$$

$$Z = 1 + \sum_{i=1}^{L-1} \gamma^i, \quad (6)$$

where Z is the normalizer to avoid the scale of the elements in h_f and h_b growing too big for longer words. The final output of the BLSTM is the concatenation of h_f and h_b. The decay factor γ ranges between 0 and 1. When γ equals 0, this method is equivalent to the conventional one. And when γ is taken to 1, the output is a simple average of all outputs.

3.2. Position-based methods

The 1-dimensional position-based method is similar to the decay based method. The difference is that we learn the best setting of the weightings rather than make it a hyper-parameter. With $w_f = (w_{f,1}, w_{f,2}, ..., w_{f,l_c}) \in \mathbb{R}^{l_c}$ and $w_b = (w_{b,1}, w_{b,2}, ..., w_{b,l_c}) \in \mathbb{R}^{l_c}$ representing the learned weightings for the forward LSTM and backward LSTM respectively, the final outputs are calculated by the following equations:

$$h_f = \frac{H_{f,t} w_f^T}{Z} = \frac{\sum_{i=1}^{L} w_{f,i} h_{f,i}}{\sum_{i=1}^{L} w_f}, \tag{7}$$

$$h_b = \frac{\sum_{i=1}^{L} w_{b,L-i+1} h_{b,i}}{\sum_{i=1}^{L} w_b}, \tag{8}$$

We also proposed a 2-dimensional position-based weighting method. This is inspired by the assumption that each dimension of the uni-direction LSTM's output is a small feature extractor, and that the different extractors shouldn't share the same positional weighting. Therefore, weighting matrices $W_f \in \mathbb{R}^{n \times l_c}$ and $W_b \in \mathbb{R}^{n \times l_c}$ rather than simply two weighting vectors are used here for computing the weighted sum of the final outputs. The calculation can be represented as the following equations:

$$h_f(i) = \frac{H_{f,t}(i,*) W_f(i,*)^T}{\sum_{j=1}^{L} W_f(i,j)}, 1 \leqslant i \leqslant L \tag{9}$$

$$h_b(i) = \frac{H_{b,t}(i,*) W_f(L+1-i,*)^T}{\sum_{j=1}^{L} W_f(L+1-i,j)}, 1 \leqslant i \leqslant L \tag{10}$$

where $X(i,*)$ refers to the i-th row of matrix X. These two formulations are implemented by element-wise multiplication of W and H and normalization by the sum of each row in W.

3.3. Attention-based methods

Attention mechanism has been successfully applied in many tasks, including question answering[14, 15], machine translation[16, 17, 18] and acoustic modelling[19]. In this section, we use two commonly used variations of self-attention to decide the weightings of the columns in $H_{f,t}$ and $H_{b,t}$. The detailed calculations for the forward LSTM are showed as follows:

$$\alpha = softmax(v^T tanh(W_f H_{f,t} + b)) \tag{11}$$

or

$$\alpha = softmax(w_f H_{f,t}) \tag{12}$$

$$h_f = \alpha H_{f,t} \tag{13}$$

where $w_f \in \mathbb{R}^n$, $W_f \in \mathbb{R}^{n*k}$, $b \in \mathbb{R}^k$ and $v \in \mathbb{R}^k$ are trained parameter vectors. The final output h_b is calculated the same way.

4. Experiments

4.1. Dataset and Experimental Setup

In this work, experiments are conducted on THUYG-20 for Uyghur and the Turkish Broadcast News Speech and Transcripts dataset for Turkish. We downloaded a large text corpus for Uyghur from a github repository[1], which was crawled from the website of China Broadcast. For Turkish, we use a collection of text in Wikipedia[2]. Detailed information and the word error rate (WER) of word level ASR for the two datasets are listed in Table 1.

Table 1: *Statistics of Uyghur and Turkish Data*

Item	Uyghur	Turkish
length of speech in training set	20h	124h
length of speech in validation set	1h	2.5h
length of speech in test set	2.4h	2.5h
#types in training transcript	17614	63299
#types in all text for training LM	101709	154798
#tokens in training transcript	108538	759279
#tokens in the large text corpus	6582657	3978078
WER for word-level ASR system	22.01	19.88

In the preprocessing stage, we convert all the Arabic characters of the Uyghur language into the 52 capital and lower English characters by the converting tool provided in the THUYG-20 dataset. And we convert all the Turkish characters into lower letters.

As described in introduction, for both Uyghur and Turkish, the spelling of a word is highly related with its pronounciation. For both languages, the lexicon used in ASR systems are generated by splitting the words and morphs into letters and treat each character as a phone. The acoustic models are both 4-layer DNNs with 1024 units in each hidden layer. 3-gram language models trained by SRILM[20] are used in building the word level 1-pass ASR systems for both languages, while 4-gram language models are used for morph level 1-pass systems. The acoustic model and the decoding progress are conducted with Kaldi speech recognition tool-kit[21].

4.2. morphological Segmentation

To deal the heavy data sparsity problem, the Morfessor toolkit[22] is used for the morphologically parsing for the two languages. We add "+" signs before the morphs which are not the first morph of a word. The aim of this operation is to make it easier to turn morph sequences into words. In our previous study, we conclude that the consistency between modeling units for the ASR system and neural language model is crucial to the performance of the rescoring process. We first change the ratio of the two items in the loss of MDL to find the best unit collection. Here, we use the mixture of the training transcripts and the large text corpus to train the segmentation model for Uyghur. While as for Turkish, the segmentation model trained only by the training transcripts performs better. That may be because the topic of the text corpus is much different from the train and test transcripts. WER results for the unit selecting experiments are listed in Table 2.

We choose the models with 43382 units and 8871 units for the Uyghur and Turkish datasets respectively. The best choices are very different for the two languages. Besides the differences in characteristics between the two languages, the scale of the training data set might affect the best choice for the number of units. The selected units are also used for neural language modeling. While WERs are still computed on the original word level.

[1] https://github.com/azmat21/UyghurTextResource

[2] https://dumps.wikimedia.org/trwiki

Table 2: *Experiments on unit selection of Uyghur and Turkish*

Uyghur		Turkish	
#units	WER(%)	#units	WER(%)
6535	21.32	6460	18.45
9212	20.95	8006	18.25
11676	20.81	8871	18.23
14095	20.61	13356	18.23
24608	20.64	16680	18.37
24608	20.64	24656	18.44
43382	19.98	44681	18.58
48259	20.05	66049	18.53

4.3. Neural Network Language Model Results

In this section, neural language models are tested on both languages by perplexity (PPL) and WER. In the Table3, "charCNN" refers to the method introduced in Section 2.2 and "charBLSTM_end" refers to the one introduced in Section 2.3. The rest items refer to the corresponding methods introduced in Section 3. The influence of the decay factor γ is presented in Figure 3 and Figure 4 for the two languages respectively.

Table 3: *Experiments on different language models*

language	model	parameter	PPL	WER
Uyghur	1-pass(4-gram)	-	238.67	19.98
	vanilla LSTM	233.7M	262.17	18.14
	charCNN	160.6M	249.79	17.90
	charBLSTM_end	138.1M	209.86	17.86
	charBLSTM_decay(0.9)	138.1M	200.22	17.60
	charBLSTM_decay(0.3)	138.1M	186.41	17.79
	charBLSTM_ave	138.1M	205.42	17.71
	charBLSTM_pos	138.1M	232.19	18.15
	charBLSTM_pos2d	138.1M	212.11	17.92
	charBLSTM_att	138.3M	217.46	18.12
	charBLSTM_att2	138.1M	221.54	18.11
Turkish	1-pass(4-gram)	-	53.07	18.23
	vanilla LSTM	61.2M	34.71	16.46
	charCNN	72.5M	33.15	16.33
	charBLSTM_end	70.4M	34.86	16.41
	charBLSTM_decay(0.9)	70.4M	34.71	16.27
	charBLSTM_decay(0.5)	70.4M	34.52	16.35
	charBLSTM_ave	70.4M	34.96	16.38
	charBLSTM_pos	70.4M	32.66	16.36
	charBLSTM_pos2d	70.5M	33.11	16.39
	charBLSTM_att	71.2M	34.68	16.41
	charBLSTM_att2	70.4M	34.70	16.31

When re-scoring, we enumerate the β in Eq.(3) from 0.0 to 1.0 with a step of 0.1. The best choice β for Uyghur and Turkish are both around 0.6. As shown in the figures and the table, better perplexity results don't always lead to better WER results. This has also been reported by many works. A possible reason is that lower perplexity means the right words get higher probabilities by the language model, while the models that could better distinguish the correct words and other confusable words do better on reducing recognition errors.

For both languages, "charBLSTM_decay" models yield the best WER and the best decay factor choice are both 0.9. While for lower perplexities, 0.3 and 0.5 are better choices for the decay factor for the two languages respectively. The performance for position-based or attention-based methods are quite unstable. As for Turkish, position-based methods result in much lower perplexities relative to "charBLSTM_end" and the attention-based method, carried out by Eq.(12), leads to a lower WER result. While the performance for both the position and attention-based methods are worse than the conventional "charBLSTM_end".

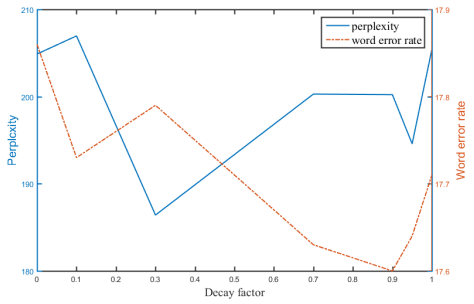

Figure 3: *The influence of the decay factor γ for Uyghur*

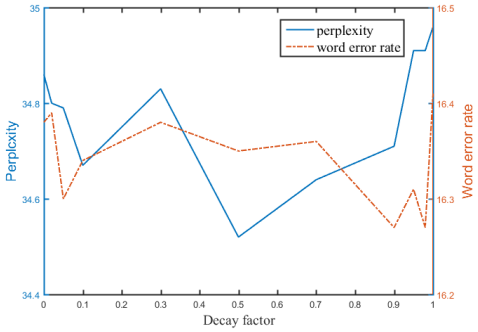

Figure 4: *The influence of the decay factor γ for Turkish*

5. Conclusions

In this paper, we first applied the statistical morphological segmentation and discussed the number of morphs for the two languages. The best number of sub-word is 43382 and 8871, which yield 2.03% and 1.65% WER reduction from word level systems for Uyghur and Turkish respectively. For both languages, the decay based weighting methods lead to the most significant WER reduction of 2.38% and 1.96% from the sub-word level 1-pass ASR system respectively. The position and attention-based methods behave unstable. For Turkish, the position-based methods work well on PPL and the attention-based methods work well on WER. While they both can't lead to better performance for Uyghur.

6. Acknowledgements

This work is partially supported by the National Key Research and Development Program (Nos. 2016YFB0801203, 2016YFB0801200), the National Natural Science Foundation of China (Nos. 11590774, 11590770),the Key Science and Technology Project of the Xinjiang Uygur Autonomous Region (No.2016A03007-1),the Pre-research Project for Equipment of General Information System (No.JZX2017-0994/Y306).

7. References

[1] H. Sak, M. Saraçlar, and T. Gungor, "Morpholexical and discriminative language models for turkish automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 8, pp. 2341–2351, 2012.

[2] M. Creutz, T. Hirsimäki, M. Kurimo, A. Puurula, J. Pylkkönen, V. Siivola, M. Varjokallio, E. Arisoy, M. Saraçlar, and A. Stolcke, "Morph-based speech recognition and modeling of out-of-vocabulary words across languages," *ACM Transactions on Speech and Language Processing (TSLP)*, vol. 5, no. 1, p. 3, 2007.

[3] H. Sak, M. Saraclar, and T. Güngör, "Morphology-based and sub-word language modeling for turkish speech recognition," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5402–5405.

[4] K. Kirchhoff, D. Vergyri, J. Bilmes, K. Duh, and A. Stolcke, "Morphology-based language modeling for conversational arabic speech recognition," *Computer Speech & Language*, vol. 20, no. 4, pp. 589–608, 2006.

[5] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[6] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5528–5531.

[7] Y. Wu, T. He, Z. Chen, Y. Qian, and K. Yu, "Multi-view lstm language model with word-synchronized auxiliary feature for lvcsr," in *CCL*, 2017.

[8] P. Bojanowski, A. Joulin, and T. Mikolov, "Alternative structures for character-level rnns," *arXiv preprint arXiv:1511.06303*, 2015.

[9] W. Ling, T. Luís, L. Marujo, R. F. Astudillo, S. Amir, C. Dyer, A. W. Black, and I. Trancoso, "Finding function in form: Compositional character models for open vocabulary word representation," *arXiv preprint arXiv:1508.02096*, 2015.

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

[11] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[12] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models." in *AAAI*, 2016, pp. 2741–2749.

[13] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2016, pp. 207–212.

[14] M. Tan, C. d. Santos, B. Xiang, and B. Zhou, "Lstm-based deep learning models for non-factoid answer selection," *arXiv preprint arXiv:1511.04108*, 2015.

[15] H. Xu and K. Saenko, "Ask, attend and answer: Exploring question-guided spatial attention for visual question answering," in *European Conference on Computer Vision*. Springer, 2016, pp. 451–466.

[16] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Advances in Neural Information Processing Systems*, 2015, pp. 1693–1701.

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[18] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[19] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[20] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *Seventh international conference on spoken language processing*, 2002.

[21] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[22] P. Smit, S. Virpioja, S.-A. Grönroos, M. Kurimo *et al.*, "Morfessor 2.0: Toolkit for statistical morphological segmentation," in *The 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Gothenburg, Sweden, April 26-30, 2014*. Aalto University, 2014.