Word and Class Common Space Embedding for Code-switch Language Modelling

Conference Paper · February 2019

DOI: 10.1109/ICASSP.2019.8683678

2 authors:

Grandee Lee
National University of Singapore
12 PUBLICATIONS **45** CITATIONS
SEE PROFILE

Haizhou Li
National University of Singapore
817 PUBLICATIONS **11,614** CITATIONS
SEE PROFILE

WORD AND CLASS COMMON SPACE EMBEDDING FOR CODE-SWITCH LANGUAGE MODELLING

Grandee Lee, Haizhou Li

National University of Singapore, Singapore
grandee.lee@u.nus.edu, haizhou.li@nus.edu.sg

ABSTRACT

Code-switch language modelling is challenging due to limited linguistic resources and less predictable word sequences. Many state-of-the-art systems rely on linguistic information such as Part-of-Speech (POS) or classes to generalize the lexicon. Such systems generally use multi-task learning or conditional network to improve over baseline RNN language model by providing a better word prediction. To overcome the data sparsity through continuous space modelling and back-off mechanism, we propose to constrain the word and class embedding in a common space by means of cross-lingual word embedding, and to make use of the predicted class embedding as a back-off scheme when word prediction model is weak. The proposed word and class Common Space embedding Language Model (CSLM) is able to model word prediction better and is more robust when only sparse training data are available. The CSLM outperforms the state-of-the-art language model by 9.7% on the code-switch SEAME corpus.

Index Terms— code-switch, language modelling, cross-lingual embedding

1. INTRODUCTION

Code-switching or code-mixing is a phenomenon in writing or conversation where two language systems are mixed within a sentence, i.e. intra-sentential, or between two sentences, i.e. inter-sentential. This is an increasingly common linguistic behavior amongst bilingual speakers [1]. Intra-sentential code-switch speech poses a significant challenge to ASR systems [2]. This is because code-switch introduces more vocabulary choices at each prediction step due to words from another language, at the same time, it occurs sparingly and freely without adhering to rigid syntactic or grammatical rules [3]. Speaker may choose when to and not to switch given the same preceding context.

The challenge is further exacerbated because there are far less code-switch linguistic resources than monolingual ones. In general, we rely on large text corpus in written form, such as newspapers and books, for monolingual language modelling. As code-switch takes place mostly in spoken form, we cannot find as much documented code-switch text as monolingual text for language modelling.

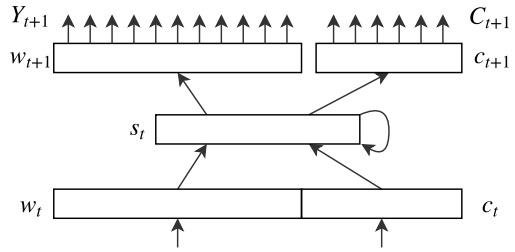
Many existing solutions leverage on linguistic information such as POS tags [4], language ID [5,6] and word classes [7] or its combination [8–10] to further generalize the lexicon, under the assumption that a generalized representation of the word sequence should be observed in the training set or to serve as an indicator for better prediction. Others [11–13] seek to detect code-switch points. Code-switch permission constraints [14, 15] are also used to incorporate code-switch probability into the language model. This has been shown to improve the WER on ASR systems, however, it is also pointed out that speakers, under a natural setting, may not abide by the proposed linguistic constraints [3]. In [16–18], the code-switch language model is improved with additional artificially generated code-switch data.

To overcome the data sparsity problem, people have attempted three broad categories of techniques. 1) Extending the input dimension by concatenating lexicon with other auxiliary features c_t such as class or POS tags, as shown in the input layer of Fig. 1(a) and Fig. 1(b). 2) Utilizing multi-task learning by making use of additional supervisory signal C_{t+1} such as class, POS tag or code-switch point labels, as illustrated in the output layers of Fig. 1(a) and Fig. 1(b). This serves as a scaffold for better generalization and learning. 3) Sharing of latent layers s_t, i.e another auxiliary network which is trained to optimize on POS tag prediction could share its last or intermediate layer to the word prediction network as a form of information transfer. Illustrated in Fig. 1(b), latent layer s_t^C for predicting class c_{t+1} is transferred to the word prediction network [19]. A combination of such techniques can achieve performance superior to a baseline RNN based language model [20] in which the probability of predicting the next word is simply conditioned upon the hidden recurrent output, $p(w_{t+1}|w_{<t}) = p(w_{t+1}|s_t)$.

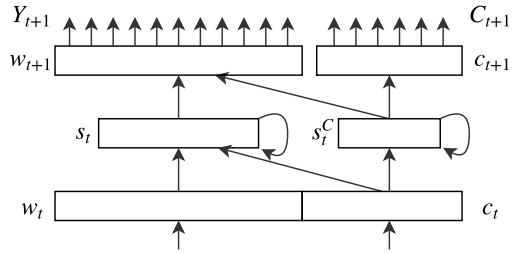
When the language model is jointly trained on input w_t and c_t, we can understand the model mathematically as conditioning the probability of the predicted word given the word history $p(w_{t+1}|w_{<t})$ with the auxiliary class c_t,

$$p(w_{t+1}|w_{<t}) = p(w_{t+1}|c_t, s_t), \tag{1}$$

in which s_t is the hidden recurrent state that contains the history for both previous lexicon and auxiliary class. However,



(a) Class based neural language model.



(b) Multi-task language model with sharing of latent layers.

Fig. 1. Architectures for multi-task and auxiliary feature based approaches. w is the vector for word and y is the output vector before softmax. Y and C are the one-hot label for lexicon and class respectively.

this formulation, that predicts w_{t+1} with c_t, departs from the statistical class-based language model [21], in which we are predicting the next word w_{t+1} conditioned on the next auxiliary class c_{t+1}.

$$p(w_{t+1}|w_{<t}) = p(w_{t+1}|c_{t+1})p(c_{t+1}|c_{<t}) \quad (2)$$

Due to this difference in formulation, the prediction of the word w_{t+1} in Fig. 1 could not benefit directly from the prediction of the auxiliary class c_{t+1} which could serve as a back-off when the class sequence is observed but not the word sequence. This paper seeks to address this problem by combining the strength of auxiliary class and back-off scheme as summarized in the formulation below.

$$p(w_{t+1}|w_{<t}) = p(w_{t+1}|w_{<t}, c_{t+1})p(c_{t+1}|c_{<t}) \quad (3)$$

We realize the above formulation using a recurrent neural network and would expect the predicted c_{t+1}, which is more reliable than word prediction with limited data, to provide stronger back-off to the word prediction network. We implement this model through 1) sharing a common embedding space which allows for an architectural improvement over the traditional multi-task based language model and at the same time 2) using the predicted embedding of the auxiliary class c_{t+1} as the input to the word prediction network together with w_t.

2. COMMON SPACE LANGUAGE MODELLING

The main difference between the proposed CSLM in Fig. 2 and the models illustrated in Fig. 1 is the usage of output from

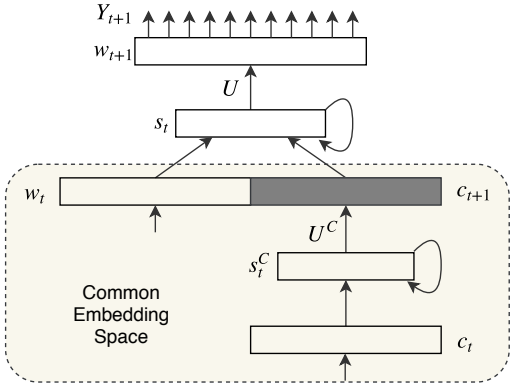


Fig. 2. Common Space embedding and class back-off Language Model (CSLM). U and U^C are the projection matrices.

the class prediction as a back-off information to the word prediction Y_{t+1}, represented by the shaded box. The main challenge in realizing the network illustrated in Fig. 2, is the discrepancy in the vector space of the output vector c_{t+1} and the input word embedding w_t. Since, traditionally, the output label for c_{t+1} is the auxiliary class label C_{t+1}, that is represented in a different embedding space than w_t. Thus by passing this embedding to the input of the word prediction network, the network has to learn another transformation matrix for the class model. This limits the useful information transferred and in the same time embedding information is lost.

In this model we choose to predict the class embedding of the next class, i.e. c_{t+1} directly so that the input and output of the auxiliary class prediction reside in the same embedding space. Consequently, the input embedding w_t also resides in the same embedding space by using pre-trained cross-lingual word embedding.

2.1. Cross-lingual Word Embedding

Besides the common space between words and classes, it is also beneficial to encode the words in both code-switching languages in the same embedding space. The cross-lingual word embedding is derived using the same technique present in BiSkip [22], whereby embeddings are derived from aligned parallel sentences. Resultant word embeddings of related words are grouped together and at the same time, the syntactic information is preserved. This is done by extending the Skip-gram [23] model which predicts the context using the word of interest, to jointly predict the context in the aligned languages and vice versa. Thus, taking English and Chinese as l_1 and l_2 respectively, the embedding will preserve the quality of the monolingual embedding (i.e. $l_1 \rightarrow l_1, l_2 \rightarrow l_2$) and at the same time tie the two monolingual embedding together using the constraint of its counterpart language (i.e. $l_1 \rightarrow l_2, l_2 \rightarrow l_1$). Using pre-trained cross-lingual word embedding in the language model also implies that we have incorporated information from the large parallel corpus. This

has a two-fold effect, firstly, the embedding for individual words will be more representative since each word will be learned with more context. Secondly, rare words in the SEAME training set which some are only seen once will benefit from these large corpora.

2.2. Back-off Class Embedding

The pre-trained cross-lingual word embedding is used for w_t,

$$w_t = vec_{Biskip}(word_t), \tag{4}$$

$$c_t = k\text{-}means(w_{t \in V}), \tag{5}$$

and the same embedding space is used for c_t by k-means clustering of all word embeddings in the vocabulary V. In CSLM, the label for the input c_t is the vector c_{t+1} representing the centroid of one of the k-means clusters.

We expect that the class embedding prediction has a profound impact on the word prediction, because the class embedding naturally contains more information than the label, due to denser representation. As it is being predicted directly, ensuring less embedding information loss than predicting only the class label. The class embedding provides the back-off to the unseen or rare word sequences, of which the class sequences have been observed in the training set. In the case that the correct class is being predicted for the corresponding word lexicon, a more informative input is being provided for the model and thus greatly reducing the perplexity.

3. EXPERIMENT

The parallel corpora from OpenSubtitle [24] and TedTalk [25] are used for training cross-lingual word embeddings. The combined corpus comprises $147M$ tokens and a total vocabulary of $(51 + 260)K$, for Chinese and English respectively. The corpus covers 92% of the SEAME English vocabulary and 100% Chinese vocabulary. SEAME Train set is added to the corpus to complete the coverage for English vocabulary. The corpus used for training the language model is SEAME (South-East Asia Mandarin-English) corpus [26], which is a spontaneous conversational code-switch corpus recorded under the setting of casual conversation or interview. For the purpose of language modelling, we use the audio transcription. In the pre-processing step, hesitation, paralinguistic markers and punctuations are removed and the Mandarin text is segmented [27]. The composition of the data set is summarized in Table 1.

Table 1. SEAME Phase II Database. Switch Point Fraction is the average code-switch points over word boundaries per utterance, it indicates complexity of the code-switch text.

	Train set	Dev set	Eval set
Tokens	$1.2M$	$65K$	$60K$
SPF	0.23	0.23	0.23
Vocabulary (CN+EN)	$(8 + 17)K$	–	–

We choose 300 dimensions for the word embedding and 200 clusters for the class embedding. The embedding space is initialized using the aforementioned pre-trained word embedding covering the vocabulary of the SEAME Train set, and made non-trainable. We use LSTM [28] which has been the state of the art in neural language modelling. The recurrent layers for both *lexicon* and *class* are set to be 2 and the hidden state dimensions 600 and 300 respectively following the embedding dimension. The number of layers and the dimensions are also comparable to the previous benchmarks, in which a similar LSTM model [19] is used with 2 recurrent layers and 500 hidden dimensions.

$$y_{t+1} = U(LSTM_{lexicon}([w_t; c_{t+1}])) \tag{6}$$

$$c_{t+1} = U^C(LSTM_{class}(c_t)) \tag{7}$$

Additionally, drop-out of 0.4 is applied in between recurrent layers and not in between recurrent time steps, to force the preceding recurrent layers to provide more robust representations and at the same time preserve the information flow between recurrent neurons [29]. For the auxiliary class prediction, MSE is used as the loss function and for lexicon prediction, cross entropy is used. The cross entropy loss is,

$$Loss = -\frac{1}{N} \sum_{i=t+1}^{N} Y_i \ln(p_i), \tag{8}$$

$$p_i = \frac{e^y_{t+1}}{\sum_j^{Vocab} e^{y_j}}. \tag{9}$$

In Equation 8, Y_i is the ground truth one-hot vector, P_i is the normalized probability of the model to predict each word in the vocabulary and N is the number of time steps. Since Tensorflow[1], by default, computes the cross entropy in nat rather than in bit, the perplexity is, $PPL = e^{Loss}$.

3.1. Results

To gain a better understanding of the effect of each proposed improvements, we conducted an ablation test to single out the perplexity contribution of each change. The first experiment in Table 2 reports the results of Class LSTM presented in Fig. 1(a) as the baseline. The inputs are word and class label of the current time. In the next two experiments, we implement CSLM with the pre-trained word and class embeddings. To illustrate the idea of using c_{t+1} as back-off is better than using c_t as the auxiliary input along with w_t, the shaded box (refer to Fig. 2) of CSLM+$c_{t,oracle}$ is replaced with the true class embedding of c_t, while for CSLM+$c_{t+1,oracle}$ it is replaced with true class embedding of c_{t+1}. Results show that if we can accurately predict the class embedding, c_{t+1} is more useful than c_t. Because a strong prior is provided for the word prediction and the model is less confused on the next word, similar to a statistical class-based language model.

[1] https://www.tensorflow.org

To show the effect of word class common space embedding against unconstrained embedding space, the next two experiments differ only in the output label for the auxiliary class prediction network. The former uses one-hot class *Label* which means the output embedding space is not constrained, the latter uses class embedding (i.e. CSLM) which constrains the embedding space to be same as the word embedding. Since the class label does not place a constraint on the output vector of the predicted class, the information shared is less useful and also the auxiliary model may lose some valuable information in maximizing the prediction accuracy of the class label. Only when the target vector of the next class is used, does the model achieve the greatest perplexity reduction.

Table 2. Ablation Analysis for CSLM

Model	PPL Dev	PPL Test
Class LSTM	144.27	143.50
CSLM + $c_{t,oracle}$	141.18	143.43
CSLM + $c_{t+1,oracle}$	5.01	5.91
CSLM + *Label*	141.18	143.43
CSLM	128.12	129.85
+ Multi-task	128.54	128.02

This also demonstrates the effectiveness of predicting the auxiliary class first then the word as expressed in Equation 3. More specifically, this model computes the probability of the next word based on the word history as well as the predicted class. Thus, the model is more robust in generalizing for unseen sequences in the testing phase as long as an appropriate class embedding could be provided for the input word. Further multi-task objective provides less improvement which confirms that the class information is being efficiently transferred from the auxiliary network. We achieve 10.7% perplexity reduction over the baseline.

4. EVALUATION

To show that CSLM tackles the problem presented by code-switch text, we compare the improvement in perplexity of CSLM to Class LSTM model across different segment of the SEAME Eval set. The perplexity reduction in the code-switch word sequences of the Eval set is 24.5%, while the reduction over monolingual word sequences is 18.2%. It is expected that the model will also improve the monolingual segment, however, since the code-switch word sequences suffer more from data sparsity, the CSLM model is more effective to code-switch word sequences than to monolingual ones.

Table 3. Perplexity for different language segments in Eval set compared to baseline class-based LSTM.

Model	Code-switch	Monolingual
Class LSTM	196.94	138.00
CSLM	148.63	112.91

It should be noted that the improvement in the perplexity of LSTM and Multi-task [19] language model in Table 4 is in part due to the increase in the training set over previous models which are carried out on SEAME Phase I. Table 4 summarize the improvements in perplexity for various combination of aforementioned techniques. There is 9.7% perplexity reduction between *CSLM + Multi-task* over the previous state-of-the-art code-switch language model *Multi-task* in Table 4. CSLM draws strength from pre-trained cross-lingual word embedding, the successful sharing of this information from the auxiliary class prediction to the lexicon prediction with minimal loss and providing strong back-off. Compared to previous models which use multi-task objective, there is generally lack of strong back-off scheme. Without a common embedding space, the lexicon model would have to solve the additional task of feature extraction since the useful feature from auxiliary class and lexicon embedding are from different space.

Table 4. Language model baseline on SEAME test set. Models marked with † indicate that training and testing are done on SEAME Phase I which approximate to 60% of SEAME Phase II in term of total tokens. Models marked with * indicate training and testing done on SEAME Phase II.

Model	PPL Dev	PPL Eval
RNNLM† [4]	246.60	287.88
FL + OF† [4]	219.85	239.21
FLM† [10]	177.79	192.08
LSTM* [19]	150.65	153.06
Multi-task* [19]	141.86	141.71
CSLM	128.12	129.85
CSLM + Multi-task	128.54	128.02

5. CONCLUSION

The experiments support the proposed CSLM in alleviating the sparsity issue in code-switch language modelling. The ablation analysis justifies the claim of using predicted class embedding c_{t+1} as a back-off to word prediction and constraining a shared embedding space between the word input and class output. We achieve 10.7% perplexity reduction over the baseline and 9.7% perplexity reduction over the previous state-of-the-art model. We show that the improvement in code-switch word sequence of the SEAME Eval set is greater, making this method a good way for code-switch language modelling.

6. ACKNOWLEDGMENT

This research is supported by National Research Foundation through the AI Singapore Programme, the AI Speech Lab: Automatic Speech Recognition for Public Service Project AISG-100E-2018-006. Grandee Lee's research is also supported by the NUS Research Scholarship.

7. REFERENCES

[1] Pieter Muysken, Carmen Pena Díaz, Pieter Cornelis Muysken, et al., *Bilingual speech: A typology of code-mixing*, vol. 11, Cambridge University Press, 2000.

[2] Özlem Çetinoğlu, Sarah Schulz, and Ngoc Thang Vu, "Challenges of computational processing of code-switching," in *Proceedings of the Second Workshop on Computational Approaches to Code Switching*. 2016, pp. 1–11, Association for Computational Linguistics.

[3] Peter Auer, "From codeswitching via language mixing to fused lects: Toward a dynamic typology of bilingual speech," *International journal of bilingualism*, vol. 3, no. 4, pp. 309–332, 1999.

[4] Heike Adel, Ngoc Thang Vu, Franziska Kraus, Tim Schlippe, Haizhou Li, and Tanja Schultz, "Recurrent neural network language modeling for code switching conversational speech," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 8411–8415.

[5] Khyathi Chandu, Thomas Manzini, Sumeet Singh, and Alan W Black, "Language informed modeling of code-switched text," in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, 2018, pp. 92–97.

[6] Younes Samih, Suraj Maharjan, Mohammed Attia, Laura Kallmeyer, and Thamar Solorio, "Multilingual code-switching identification via LSTM recurrent neural networks," in *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, 2016, pp. 50–59.

[7] Zhiping Zeng, Haihua Xu, Tze Yuang Chong, Eng-Siong Chng, and Haizhou Li, "Improving n-gram language modeling for code-switching speech recognition," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2017, pp. 1596–1601.

[8] Ngoc Thang Vu and Tanja Schultz, "Exploration of the impact of maximum entropy in recurrent neural network language models for code-switching speech," in *Proceedings of The First Workshop on Computational Approaches to Code Switching*, 2014, pp. 34–41.

[9] Heike Adel, Ngoc Thang Vu, Katrin Kirchhoff, Dominic Telaar, and Tanja Schultz, "Syntactic and semantic features for code-switching factored language models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 431–440, 2015.

[10] Heike Adel, Ngoc Thang Vu, and Tanja Schultz, "Combination of recurrent neural networks and factored language models for code-switching language modeling," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2013, vol. 2, pp. 206–211.

[11] Emre Yılmaz, Henk van den Heuvel, and David van Leeuwen, "Code-switching detection using multilingual DNNs," in *Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 610–616.

[12] Emre Yılmaz, Henk van den Heuvel, and David A van Leeuwen, "Acoustic and textual data augmentation for improved ASR of code-switching speech," *arXiv preprint arXiv:1807.10945*, 2018.

[13] Thamar Solorio and Yang Liu, "Learning to predict code-switching points," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2008, pp. 973–981.

[14] Ying Li and Pascale Fung, "Improved mixed language speech recognition using asymmetric acoustic model and language model with code-switch inversion constraints," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 7368–7372.

[15] Ying Li and Pascale Fung, "Language modeling with functional head constraint for code switching speech recognition," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 907–916.

[16] Ngoc Thang Vu, Dau-Cheng Lyu, Jochen Weiner, Dominic Telaar, Tim Schlippe, Fabian Blaicher, Eng-Siong Chng, Tanja Schultz, and Haizhou Li, "A first speech recognition system for mandarin-english code-switch conversational speech," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4889–4892.

[17] Ewald van der Westhuizen and Thomas R Niesler, "Synthesised bigrams using word embeddings for code-switched ASR of four south african language pairs," *Computer Speech & Language*, vol. 54, pp. 151–175, 2019.

[18] Ewald van der Westhuizen and Thomas Niesler, "Synthesising isizulu-english code-switch bigrams using word embeddings," *Proc. Interspeech 2017*, pp. 72–76, 2017.

[19] Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung, "Code-switching language modeling using syntax-aware multi-task learning," in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*. 2018, pp. 62–67, Association for Computational Linguistics.

[20] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur, "Recurrent neural network based language model," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010, pp. 2877–2880.

[21] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai, "Class-based n-gram models of natural language," *Computational linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[22] Thang Luong, Hieu Pham, and Christopher D Manning, "Bilingual word representations with monolingual quality in mind," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, 2015, pp. 151–159.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[24] Pierre Lison and Jörg Tiedemann, "OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC)*, 2016, pp. 923–929.

[25] Jrg Tiedemann, "Parallel data, tools and interfaces in OPUS," in *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, 2012, pp. 2214–2218.

[26] Grandee Lee, Thi-Nga Ho, Eng-Siong Chng, and Haizhou Li, "A review of the Mandarin-English code-switching corpus: SEAME," in *Asian Language Processing (IALP), 2017 International Conference on*. IEEE, 2017, pp. 210–213.

[27] Pi-Chuan Chang, Michel Galley, and Christopher D Manning, "Optimizing chinese word segmentation for machine translation performance," in *Proceedings of the third workshop on statistical machine translation*. Association for Computational Linguistics, 2008, pp. 224–232.

[28] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney, "LSTM neural networks for language modeling," in *Thirteenth annual conference of the international speech communication association*, 2012, pp. 194–197.

[29] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.