Exploring Context's Diversity to Improve Neural Language Model

Yanchun Zhang[†,‡], Xingyuan Chen[‡] ✉
[†]*School of Computer and Software Engineering*
Xihua University
Chengdu, China
Email(Yanchun Zhang): 834612949@qq.com

Peng Jin[‡] ✉, Yajun Du[†]
[‡]*School of Computer Science*
Leshan Normal University
Leshan, China
Email(Peng Jin): jandp@pku.edu.cn

***Abstract*—The neural language models (NLMs), such as long short term memery networks (LSTMs), have achieved great success over the years. However the NLMs usually only minimize a loss between the prediction results and the target words. In fact, the context has natural diversity, i.e. there are few words that could occur more than once in a certain length of word sequence. We report the natural diversity as context's diversity in this paper. The context's diversity, in our model, means there is a high probability that the target words predicted by any two contexts are different given a fixed input sequence. Namely the softmax results of any two contexts should be diverse. Based on this observation, we propose a new cross-entropy loss function which is used to calculate the cross-entropy loss of the softmax outputs for any two different given contexts. Adding the new cross-entropy loss, our approach could explicitly consider the context's diversity, therefore improving the model's sensitivity of prediction for every context. Based on two typical LSTM models, one is regularized by dropout while the other is not, the results of our experiment show its effectiveness on the benchmark dataset.**

***Keywords*-Neural language model; Context's diversity; Loss function.**

I. Introduction

As a fundamental task, language model (LM) which uses the previous words to predict the next word has been playing an important role in natural language processing (NLP). It is widely used in speech recognition[22], machine translation[21], text generation[5] and so on. The early methods estimate n-gram probabilities of a sequence of strings (words) resting on the count-based models with counting and smoothing[12]. But they are hard to get good performance because of data sparsity. In order to alleviate the data sparsity, the neural language models (NLMs)[24][25] are proposed. The neural language models do not use the one-hot-vectors but the low-dimension vectors to represent the input sequence of the neural network. These low-dimension vectors are initialized with random parameters.

Recently, recurrent neural network (RNN)[18], a neural sequence model, has achieved the state-of-the-art in language model[19][20]. RNN like LSTM[14] is good at alleviating the problem of gradient explosion and disappearance. It becomes one of the most popular models in NLP later. To the best of our knowledge, the NLMs mentioned above usually have the only loss function between the prediction results and the target words. However, we randomly select two words from a context and will find that these two selected words are different from each other in most occasions, especially when they are close in a sequence. We show an example, a randomly selected sentence from the PTB, as follows:

sequence: *big investment banks refused to step up to the plate to support the beleaguered floor traders by buying big blocks of stock traders say heavy selling of standard & poor's 500-stock index futures in chicago unk beat stocks downward*

In this sequence, the number of the words that appear more than once is 5, and the length of the sequence is 39. The probability that the two selected words are the same is about 0.0094 which is calculated by equation (1). Equation (1) is a combinatory formula, where $E_i, i \in [1, 5]$, is the number of the $i\text{-}th$ word that appears more than once in the sequence. L is the length of the sequence and P is the probability that the two selected words are the same.

$$P = \frac{\sum_{i=1}^{5} \binom{2}{E_i}}{\binom{2}{L}} \tag{1}$$

It shows that if we randomly select two contexts in the text and feed them to train a NLM, the model's prediction results of these two contexts should be distinct. Because the target words corresponding to the two contexts are much likely to be diverse. In this paper, we introduce a new loss function that considers the diversity of the context as another constraint of the model.

Inspired by the discovery introduced above, we hope our model could improve its sensitivity of prediction for every context. In order to explicitly consider the effect brought by the diversity of different contexts, we create a new cross-entropy loss between the prediction results for any two different given contexts. The new cross-entropy loss could be added to the NLMs and is excepted to be maximized. We implement our method on the baseline then called RNNR proposed by the paper[23] - recurrent neural network regularization. The RNNR will be introduced in section III(A). The code of our model could be accessed at https://github.com/teanon/contextDiversity-LM.

In summary, our contributions are as follows:

- We observe the diversity of the text and explorit it to

improve a neural network LM without increasing model's parameters.

• Experimental results on the benchmark dataset[9] show that our method exceeds the baseline model. Our model has a decrease in the number of iteration, which will reduce the training time of a model especially that has a large size.

II. Related Work

As a fundmental task, language model is widely used in other tasks of NLP such as words prediction[13], text generation[1] and so on. So how to improve the performance of language model is particularly important. To our knowledge, many works are dedicated to modifying the network structure of the LM, such as RNN[18], the CRNNs[3][6] that conbine CNN with RNN and self-attention mechanism[17]. But the models mentioned above just use the traditional softmax loss as the model's final loss and don't try to improve their performance by modifying the loss of model.

However improving language model's performance by modifying the loss function also is an effective way[15]. [7] replaces the softmax loss with margin loss to adress the issue that softmax loss only focuses on whether the sample is correctly classified, and does not require intra-class compactness and inter-class separation. What's more, [11] adds a new loss term which attempts to equalize the probabilities of male and female words in the output to the taditional loss function to alleviate the gender bias in text generation.

In this paper, we take language model as our research task which is one kind of tasks of the RNNR. We attempt to enhance the model's sensitivity of prediction for every context via the way of adding a new cross-entropy loss function to the RNNR. Our experiments will demonstrate that our model is simple but effective.

III. The Model with the Diversity of Context

A. The Recurrent Neural Network Regularization

The RNNR is dedicated to modifying dropout operator which has been working very well in feed-forward neural networks [8][16]. Conducting dropout only on the non-recurrent connections could easily learn to conserve effective information for a long period of time. It also uses the traditional cross-entropy loss between Z and Y as the final loss for the model. As shown in equation (2), $Y_i, i \in [1, len]$ is the $i\text{-}th$ element of Y, and Y is an one-hot-vector sequence of the target words. Z_i, $i \in [1, len]$, is the softmax result corresponding to the context of the $i\text{-}th$ step in LSTM, where len is the length of the input sequence and also is the size of Z. Z is a vector sequence consisting of the softmax outputs of an entire input sequence, and it could be calculated by equation(3). The structucture of the RNNR is displayed in Figure.1, and its model is explained by the equations as follows:

$$Loss_{ZtoY} = \frac{1}{len}\sum_{i=1}^{len} crossEntropy(Y_i, Z_i) \quad (2)$$

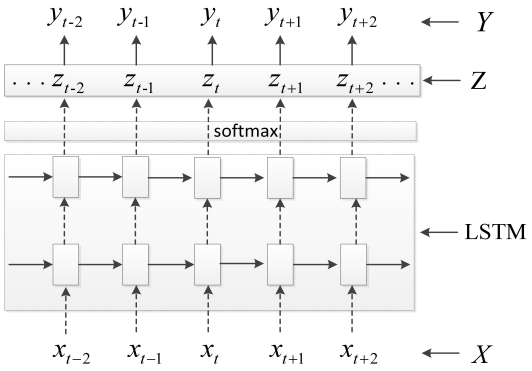

Figure 1. The dashed arrow indicates the connection to which the dropout is applied, while the solid inddicates the connection without dropout in the regularized multilayer LSTM where Z is the output sequence of softmax results.

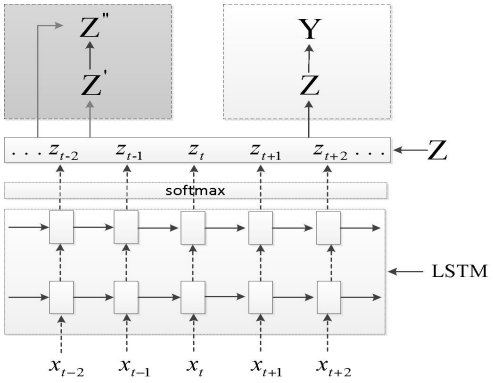

Figure 2. The architecture of our model. The dashed arrow indicates the connection to which the dropout is applied, while the solid inddicates the connection without dropout. Z is the sequence of softmax results. Both $Z^{'}$ and $Z^{''}$ are sampled from Z and our model calculates the new cross-entropy loss between these two subsets.

$$Z = softmax(LSTM(X)) \quad (3)$$

B. The Model with Context's Diversity

In our model, we add a new cross-entropy loss between $Z^{'}$ and $Z^{''}$ to the RNNR. Both $Z^{'}$ and $Z^{''}$ are sampled from Z according to some certain sampling rules and the rules will be introduced in detail in section III(C). Equation (4) is the new loss function based on the context's diversity. n is the size of $Z^{'}$ and $Z^{''}$. In our model, $Z^{'}_j$ and $Z^{''}_j$, $j \in [1, n]$, need corresponding to the softmax results of different step in LSTM. Our model is illustrated in Fig.2.

$$Loss_{Z^{'}toZ^{''}} = \frac{1}{n}\sum_{j=1}^{n} crossEntropy(Z^{'}_j, Z^{''}_j) \quad (4)$$

The new cross-entropy loss between $Z^{'}$ and $Z^{''}$ is added to the RNNR to form the final loss function shown in equation (5). The final loss function of our model is shown as follows:

$$FinalLoss = Loss_{ZtoY} + \alpha * \frac{1}{\exp Loss_{Z^{'}toZ^{''}} + \gamma} \quad (5)$$

In equation (5), α and γ are hyper-parameters. α is the weight coefficient of $Loss_{Z^{'}toZ^{''}}$ which represents the

diversity of the softmax results corresponding to n pairs of different contexts. γ is used to avoid the exception divided by zero. In our model, we want the softmax outputs of the two different contexts could vary greatly. Therefore $Loss_{Z^{'}toZ^{''}}$ needs to be maximized. Considering that the $FinalLoss$ is expected to be as small as possible, we apply the reciprocal of $Loss_{Z^{'}toZ^{''}}$ dealt with exponential function to be a part of the $FinalLoss$.

C. The Sampling Rules

Interval Sampling(ISP): In order to explore how the distance between $Z_j^{'}$ and $Z_j^{''}, j \in [1, n]$, could affect the performance of the model. We let $Z^{'} = [Z_1, Z_2, ..., Z_{len-k}]$ and then $Z^{''} = [Z_{1+k}, Z_{2+k}, ..., Z_{len}]$. Namely the step interval between $Z_j^{'}$ and $Z_j^{''}$, $j \in [1, n]$, is k in the LSTM, where k is a hyper-parameter. len is the length of the input sequence. $n = len - k$, where n is the number of elements in $Z^{'}$ and $Z^{''}$. Because the range of k is small, the change of n can be ignored.

Random Sampling(RSP): There is a conclusion, which will be explained in section V(A), that the performance of the model is not sensitive to k. For this part, we want to explore how the number of samples in $Z^{'}$ and $Z^{''}$ could affect the performance of the model. We sample $Z^{'}$ and $Z^{''}$ from Z randomly, and n is the number of samples in $Z^{'}$ and $Z^{''}$.

IV. Experiment

In order to verify the effectiveness of our model, we apply word-level prediction experiments on the PTB (Penn Tree Bank) dataset. We design two rules to sample $Z^{'}$ and $Z^{''}$ from Z, where $Z^{'}$, $Z^{''}$ and Z are introduced in section III.

We experiment on the large LSTM with dropout regularization and the small LSTM without regularization repectively. For the two sampling rules, the length of the input sequence and the batchsize are 35, 20 respectively for the unrolled large LSTM, and are 20, 20 respectively for the unrolled small LSTM. All hyper-parameters of our model are used to minimize the perplexity of the validation set. The results of the RNNR are directly cited from their paper.

A. Dataset

As the RNNR did, we also use the PTB (Penn Tree Bank) dataset to measure our model. It consists of three parts, the training set, the validation set and the test set. They are composed of 929k, 73k and 82k tokens respectively. The size of vocabulary is 10k. The dataset has been pre-processed and it contains all 10k different words. All words are lowercase and all sparse words are replaced by unk. The PTB dataset is small and easy to tain.

Table I
WORD-LEVEL PERPLEXITY ON THE PTB DATASET, THE LOWER IS THE BETTER.

Model	Validation Set	Test Set
large LSTM (RNNR)	82.2	78.4
large LSTM (ISP)	**81.2**±0.056	77.5 ±0.019
large LSTM (RSP)	81.3 ±0.001	**77.4**±0.024
small LSTM (RNNR)	120.7	114.5
small LSTM (ISP)	113.8 ±0.134	109.6 ±0.049
small LSTM (RSP)	**113.5**±0.210	**109.4**±0.477

B. The Experiment According to ISP

1) The Large LSTM : We run our model with setting the learning rate to 1 during the first 18 epochs and then we apply a factor as 0.78 to decrease the learning rate for the rest of epochs. We totally train our model with 47 epochs. For each group of parameters, we train our model five times and take the average of results as the final result. When k is set to 1 and the value of α is 0.1, the large LSTM according to ISP gets the lowest perplexity on the validation set and its perplexity on the test set drops by 0.9 compared with the RNNR, as it is shown in Table I.

2) The Small LSTM : For the first four epochs, we train our model with setting the learning rate to 1, then we use a factor that is 0.35 to decrease the leaning rate for the following epochs. The number of the total epochs is 11. We train our model five times to achive the average, serving as the final result, of the model's perplexity for each group of parameters. When the k is assigned with 5, and the α is 1, the perplexity values, displayed in Table I, that are 13.9 for the validation set and 109.7 for the test set repectively drop obviously in contrast with the RNNR.

C. The Experiment According to RSP

1) The Large LSTM : For the large LSTM, we also train our model five times to take the mean value of the model's perplexity as the final result for each group of parameters. When the hyper-parameters, except the α and n which are 5 and 20 respectively, are the same as the large LSTM introcued in section IV(B.1), the large LSTM according to RSP gains the lowest test perlexity as shown in Table I.

2) The Small LSTM : For the small LSTM, we also train our model five times to take the mean value of the model's perplexity as the final result for each group of parameters. When the small LSTM according to RSP gains the lowest perplexity values on both the validation set and the test set, as it is presented in Table I, the α is assigned with 1 and n also is set to 20 and other hyper-parameters are set to being in accordance with the small LSTM of section IV(B.2).

V. Discussion

In this part, we will mainly discuss the hyper-parameters' effect on model's performance. The hyper-parameters we focus on are k, α and n, where k introduced in ISP is the interval between $Z_j^{'}$ and $Z_j^{''}, j \in [1, n]$. n is the number of samples in $Z^{'}$ and $Z^{''}$ according to RSP. α is the weight coefficient of $Loss_{Z^{'}toZ^{''}}$.

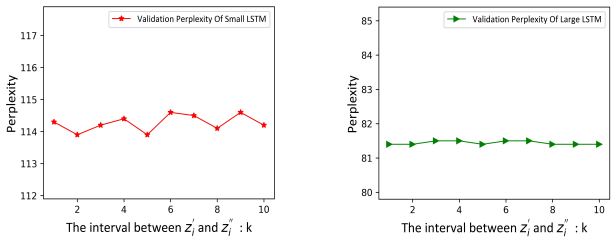

Figure 3. The k is the interval between $Z_j^{'}$ and $Z_j^{''}$ according to ISP. We vary the k from 1 to 10. With the k increasing gradually, the trends of validation perplexity for the small LSTM and the large LSTM are shown in the left picture and the right picture respectively.

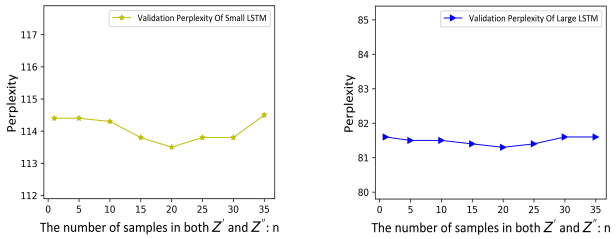

Figure 4. The x-axis represents the number of samples in both $Z^{'}$ and $Z^{''}$. The trend of validation perplexity for the small LSTM is shown in the left picture and for the large LSTM is taken on the right picture, with the increase of n.

A. The Influence of k According to ISP

At the begining, k is set to 1, and increase 1 each time until it is 10. The α of the small LSTM and the large LSTM is empirically fixed to 1 and 0.01 repectively. From the results in Fig.3, we can observe that both the large LSTM and the small LSTM have a inconspicuous fluctuating validation perplexity. The values of the variance for the small LSTM and the large LSTM are 0.06 and 0.02 respectively. So we could draw a conclusion that the performance of the model is not sensitive to k. Because the diversity of the two predicted words is not determined by their distance in the input sequence, but by the meaning of themselves and the contexts they respectively correspond to. In the same way, the diversity of two contexts also does not depend on the distance of them in the input sequence. The k used to get the lowest validdation perplexity for the small LSTM is 5, and for the Large LSTM is 1.

B. The Influence of n According to RSP

For this section, we observe the variety of performance with respect to the number of samples, i.e. n according to RSP. The α of the small LSTM and the large LSTM are empirically fixed to 1.0 and 0.1 repectively. We respectively set n to 1, 5, 10, 15, 20, 25, 30 and 35 for both the small LSTM and the large LSTM. As it is shown in Fig.4, the results of validation perplexity for these two LSTM models show the same tendency. It first decreases until n is 20 and then increases with n increasing gradually. Both the small LSTM and the large LSTM get the lowest validation perplexity when n is 20.

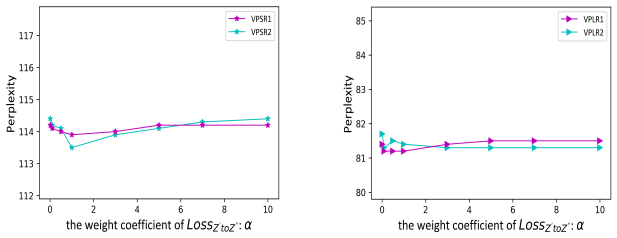

Figure 5. As it is shown in the left picture, the VPSR1 is the validation perplexity of small LSTM according to ISP, and the VPSR2 is the validation perplexity of small LSTM according to RSP. In the right picture, the VPLR1 is the validation perplexity of large LSTM according to ISP, and the VPLR2 is the validation perplexity of large LSTM according to RSP. The x-axis is the value of α which is the weight coefficient of $Loss_{Z'toZ''}$

C. The Influence of α in both ISP and RSP

The α is set to 0.01, 0.1, 0.5, 1, 3, 5, 7 and 10 repectively and the results are shown in Fig.5.

For the small LSTM whose results are shown in the left picture of Fig.5, the results of validation perplexity according to both ISP and RSP show the same tendency to decrease first and then increase with α increasing gradually in general, As it is shown in VPSR1 and VPSR2 respectively. The small LSTMs for both ISP and RSP get the lowest validation perplexity with setting the α to 1. In line with the results in section V(A) and V(B), we use the best hyper-parameters setting, i.e. k is 5 and n is 20.

For the large LSTM whose results are shown in the right picture of Fig.5, the results of validation perplexity according to both ISP and RSP change obviously when α is less than 3. But the trends for both ISP and RSP become stable gradually when α is greater than 3. As the VPLR1 shows, the large LSTM resting on ISP gets the best result with setting α to 0.1. The large LSTM based on RSP gets its best result when α is 5 as the VPLR2 shows. According to the results in section V(A) and V(B), we use the best hyper-parameters setting, i.e. k is 1 and n is 20.

VI. Conclusion

Different from previous LSTM models which only pay attention to making the prediction results to be the same as the target words as possible, our model additionally expects the prediction results of two different contexts could be different as much as possible. It is combining the $Loss_{Z'toZ''}$ with $Loss_{ZtoY}$ that our model could take the diversity of the context into consideration when the model is pursuing the consistency between the model's outputs and the labels. Considering the diversity of the context as constraint could enhance model's sensitivity of prediction. Compared with the RNNR, our model drops the test perplexity for the large LSTM by 1 and 5.1 for the small LSTM. The small LSTM and the large LSTM could get the best results after being respectively trained for 11 and 47 epochs in our model, but 13 and 55 epochs in the RNNR. So our model could reduce the training time of the model to some extent.

For the future work, we are going to explore whether the context's diversity could improve the performance

of attention-based NLM[4]. Further more, we will embed it into some applications such as neural machine tanslation[21] to further manifest the effectiveness of our method.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (61373056, 61472329, 61532006, 61602389, 6187229861874401, 61902324). Xingyuan Chen and Peng Jin are the co-crossponding authors for this paper.

REFERENCES

[1] B. Dzmitry, C. Kyunghyun and B. Yoshua, *Neural machine translation by jointly learning to align and translate*. : arXiv:1409.0473, 2014.

[2] H. B.Mcmahan, D. Ramage, K. Talwar and L. Zhang, *Learning Differentially Private Recurrent Language Models*. : international conference on learning representations, 2018.

[3] J. Chiu and E. Nichols, *Named Entity Recognition with Bidirectional LSTM-CNNs*. Transactions of the Association for Computational Linguistics, **4**, 357—370, 2016.

[4] J. Devlin, M. Chang, K. Lee and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. CoRR, abs/1810.04805, 2018.

[5] J. Guo, S. Lu, H. Cai, W. Zhang, J. Wang and Y. Yu, *Long Text Generation via Adversarial Training with Leaked Information*. : national conference on artificial intelligence, 5141—5148, 2018.

[6] J. Wang, L. Yu, K. Lai and X. Zhang, *Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model*. Berlin, Germany: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics 2016, pp. 225—230, 2016.

[7] L. Ting-En, X. Hua, *Deep Unknown Intent Detection with Margin Lossn*. : Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 5491—5496, 2019.

[8] L. Wan, M. Zeiler, S. Zhang, Y. LeCun and R. Fergus, *Regularization of Neural Networks using DropConnect*. : Proceedings of the 30th International Conference on Machine Learning (ICML-13), pp. 118—126, 2013.

[9] M. Marcinkiewicz, M. Marcus and B. Santorini, *Building a large annotated corpus of english: The penn tree bank*. Computational linguistics, **19**(2), 313—330, 1993.

[10] Q. Ai, L. Yang, J. Guo and W. Croft, *Improving Language Estimation with the Paragraph Vector Model for Ad-hoc Retrieval*. : international acm sigir conference on research and development in information retrieval, 2016.

[11] Q. Yusu, M. Urwa, Z. Ben and H.W. Jae, *Reducing Gender Bias in Word-Level Language Models with a Gender-Equalizing Loss Function*. : Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, 223—228, 2019.

[12] S. Chen, and J. Goodman, *An Empirical Study of Smoothing Techniques for Language Modeling*. : Technical Report, Harvard University, 1998.

[13] S. Ghosh, O. Vinyals, B. Strope, S. Roy, T. Dean and L. Heck, *Contextual LSTM (CLSTM) models for Large scale NLP tasks*. CoRR, abs/1602.06291, 2016.

[14] S. Hochreiter and J. Schmidhuber, *Longshort-term memory, Neural computation*. **9**(8), 1735—1780, 1997.

[15] S. Lei, X. Hu and L. Bing, *Deep open classification of text documents*. : In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2911–2916, 2017.

[16] S. Wang, and C. Manning, *Fast dropout training*. : Proceedings of the 30th International Conference on Machine Learning (ICML-13), pp. 118—126, 2013.

[17] T. Ke, B. Arianna and M. Christof, *Recurrent memory networks for language modeling*. : arXiv preprint arXiv:1601.01272, 2016.

[18] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky and S. Khudanpur, *Recurrent Neural Network Based Language Model*. : Proceedings of INTERSPEECH, 2010.

[19] T. Mikolov, S. Kombrink, A. Deoras, L. Burget and J. Cernocky, *RNNLM-recurrent neural network language modeling toolkit*. : Proceedings of the 2011 ASRU Workshop, pp. 196—201, 2011.

[20] T. Mikolov and G. Zweig, *Context dependent recurrent neural network language model*. : SLT Workshop, 2012.

[21] U. Hermjakob, Q. Li, D. Marcu, J. May, S. Mielke, N. Pourdamghani, and H. Ji, *Incident-Driven Machine Translation and Name Tagging for Low-resource Languages*. Machine Translation, 59—89, 2018.

[22] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke and G. Zweig, *The microsoft 2016 conversational speech recognition system*. : international conference on acoustics, speech, and signal processing, pp. 5255—5259, 2017.

[23] W. Zaremba, I. Sutskever and O. Vinyals, *Recurrent neural network regularization*. : arXiv preprint arXiv:1409.2329, 2014.

[24] Y. Bengio, R. Ducharme, P. Vincent and C. Janvin, *A neural probabilistic language model*. Journal of Machine Learning Research, **3**(6), 1137—1155, 2003.

[25] Y. Bengio, R. Ducharme and P. Vincent, *A Neural Probabilistic Language Model*. Journal of Machine Learning Research**3**, 1137—1155, 2003.