An Investigation on Statistical Machine Translation with Neural Language Models

Yinggong Zhao, Shujian Huang, Huadong Chen, Jiajun Chen

State Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210046, China
{zhaoyg,huangsj,chenhd,chenjj}@nlp.nju.edu.cn

Abstract. Recent work has shown the effectiveness of neural probabilistic language models(NPLMs) in statistical machine translation(SMT) through both reranking the n-best outputs and direct decoding. However there are still some issues remained for application of NPLMs. In this paper we further investigate through detailed experiments and extension of state-of-art NPLMs. Our experiments on large-scale datasets show that our final setting, i.e., decoding with conventional n-gram LMs plus un-normalized feedforward NPLMs extended with word clusters could significantly improve the translation performance by up to averaged 1.1 Bleu on four test datasets, while decoding time is acceptable. And results also show that current NPLMs, including feedforward and RNN still cannot simply replace n-gram LMs for SMT.

Keywords: statistical machine translation, neural probabilistic language model, recurrent neural network, feedforward neural network

1 Introduction

Language model is the key component of SMT system as it ensures the fluency of the translation output. For a long period, n-gram LMs, which model over discrete representations of words have been the dominant form. However data sparsity always be an obstacle of such model as the size of training data cannot meet the increasing of parameters under the one-hot representation.

To address this issue, Bengio et al. [2] proposed distributed word representations, in which each word is represented as a dense vector in a low-dimensional feature space(compared to conventional one-hot representation), modelling over which they propose a feed-forward NPLM. During training, the NPLM learns both a distributed representation for each word in the vocabulary and a probability distribution for n-grams over words representations. Furthermore, Mikolov et al. [12] introduced recurrent structure into neural network that can capture all previous words as context. Further experiments show that NPLMs can rival or even surpass traditional n-gram LMs [14, 11] evaluated in terms of perplexity.

Although recent work [17, 24] have shown that NPLMs can be directly incorporated into SMT system for direct decoding and outperform both baseline and simply re-ranking. There are some questions remained. Firstly, in both works

only un-normalized NPLM score are used during decoding. Although such usage could bring improvements, there is no detailed comparison on normalized and un-normalized NPLM for SMT decoding; secondly, current NPLM only train on part of words that occur in corpus and the rest words are simply truncated as <unk>; finally, although NPLMs can improve SMT as supplement, it would be interesting to see whether NPLMs can replace conventional n-gram LMs for SMT or other NLP tasks.

In this work, we try to answer the above questions. We demonstrate that un-normalized NPLM performs same as normalized for SMT, as context score turns to be constant, which means the cost can be heavily reduced during decoding. Then we extend current NPLMs with a simple strategy, i.e., we use word cluster for truncated words during NPLM training, which could further improve MT performance by another 0.2 BLEU on average. Finally, we compare our model with conventional n-gram LM and recurrent neural network(RNNLM) [10]. Results show that traditional n-gram LMs outperform current NPLMs, which means that they still cannot be simply replaced.

The rest of this paper is organized as follows: In section 2 we briefly review the state-of-art NPLMs with feedforward and RNN as examples. Then we present the implementation details on incorporating NPLMs to SMT systems in section 3. Experimental settings are reported in section 4. Then from 5 to 7 we discuss three questions separately. We conclude our work and present future work in section 9.

2 Overview of Neural Probabilistic Language Models

In this section, we briefly review current NPLM models. The basic idea behind NPLMs is to predict probability of word w given context $\mathbf{u}$. There are many types of NPLMs, e.g., feedforward [2], log-bilinear [13], recurrent neural network(RNN for short) [12]. One nice property of NPLMs is that for whatever n-gram input, the model can assign a non-zero score so that no smoothing or back-off is required. In this work, we mainly focus on two typical NPLMs, i.e., feedforward and recurrent.

Similar to n-gram LM, feedforward LM has fixed context size order given, whose architecture could be seen on figure 2. Let n be the order of the language model; let $\mathbf{u}$ range over contexts, i.e., strings of length $(n-1)$, and w range over words. The embeddings of $\mathbf{u}$ is concatenated at input layer and then mapped to output layer through project layer. Two hidden layers can learn non-linear knowledge. Meanwhile, the parameter complexity is $O(|V| + |H| * n)$, while it is $O(|V|^n)$ for n-gram LM.

One obstacle that hampers the application of feedforward NPLM is the heavy calculation caused by repeated summations throughout whole vocabulary under standard maximum likelihood estimation (MLE) in NPLM training. Vaswani et al. [24] combined strategies including rectifier linear units [16], noise contrastive estimation [8] and mini-batch learning for fast training.

Meanwhile, RNNLM (figure 2) is different from feedforward LM. The input layer is composed of a vector $w(t)$ that represents current word w_t and of vector $s(t-1)$ that represents hidden layer from previous step. After training, the output layer represents $P(w_{t+1}|w_t, s(t-1))$. As RNNLM also faces heavy cost of training as feedforward. Mikolov et al. [12] decomposed $P(w_{t+1}|w_t, s(t-1))$ into $P(w_{t+1}|c_{t+1}, w_t, s(t-1))P(c_{t+1}|w_t, s(t-1))$ so as to reduce complexity from $O(|V|)$ to $O(\sqrt{|V|})$.

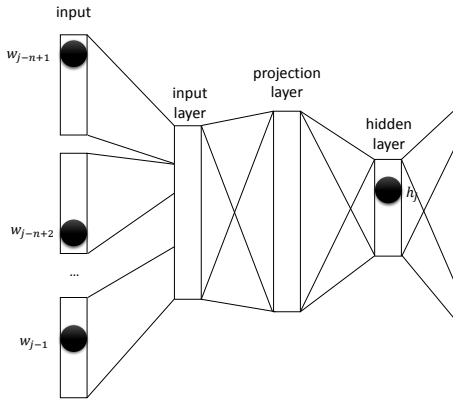

Fig. 1. Architectures of feedforward N-PLM.

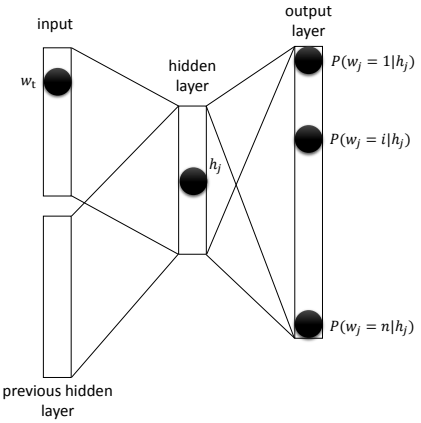

Fig. 2. Architectures of RNNLM

3 Adding NPLMs to MT System

In order to incorporate neural LMs into MT system, we first rerank k-best lists with NPLMs following previous work. As feedforward NPLM scores n-grams, it can also be integrated into decoder just as conventional n-gram models. Different from conventional n-gram LM, for which only performs search for each input query, each time matrix operation is done for NPLMs, which makes caching a necessary. During decoding, we maintain a dictionary for each thread to store the n-grams that have been calculated.

4 Experimental Setting

In this section, we first list the detailed setting for whole experiment. The training data came from the NIST 2012 Chinese-English evaluation task with sentences shorter than 60 words. Phrases were extracted from all training data, while rules with nonterminals were extracted from only the FBIS corpus (LDC2003E14).

We ran MERT on the NIST 2003 test data(MT03), while performance is tested from the NIST 2004 to 2008(referred as MT04, MT05, MT06, MT08). Alignments were extract with GIZA++ [19] with refinement from both directions. An in-house hierarchical phrase-based SMT system [5] was used to report main experimental results.

The baselines contained two conventional 5-gram LMs, estimated with modified Kneser-Ney smoothing [4] on the English side of the bitext and the 329M-word Xinhua portion of English Gigaword (LDC2011T07). Against these baselines, we tested systems that included the two conventional LMs as well as two 5-gram NPLMs trained on the same corpus. We trained both the log-linear models and the discriminative rerankers on 1000-best lists with MERT [18]. In order to make the results stable[6], we ran MERT three times and reported the average score. The results were evaluated under case-insensitive NIST Bleu.

We used nplm[1] for feedforward(ff for short) LM training. Following the setting in [24], all ff NPLMs had a vocabulary size of 100k by default while all digits are converted to 0 for training of NPLMs. The NPLMs used dimension size 150 for input and output embeddings, 750 units in hidden layer and 150 units in hidden layer. Besides, model was optimized by 10 epochs of stochastic gradient ascent(SGD) with mini-batches of size 1000 and an initial learning rate of 1. The number of noise samples was set 100 per training example. Meanwhile the RNNLMs[2] followed same setting for data as feedforward, with 150 dimensions for both word embeddings and hidden layer. Besides, 2 million direct connections under 4-gram was set for RNNLM.

5 Comparison of normalized and non-normalized NPLMs for MT

setting	mt03(dev)	mt04	mt05	mt06	mt08	test average
baseline	38.17	38.43	37.70	34.25	24.57	33.74
reranking+ff-normalized	38.84	38.62	37.98	35.00	25.08	34.17(+0.43)
reranking+ff-unnormalized	38.73	38.56	37.93	34.98	24.99	34.11(+0.37)
decoding+ff-normalized	38.91	39.45	38.72	34.87	25.27	34.58(+0.84)
decoding+ff-unnormalized	39.14	39.49	38.80	34.87	25.31	34.62(+0.88)
reranking+rnn-normalized-only	38.00	37.99	37.22	34.34	24.42	33.49 (-0.25)
reranking+rnn-unnormalized-only	37.30	36.99	35.89	33.36	23.66	32.48(-1.26)
reranking+rnn-normalized	38.69	38.46	37.85	35.04	25.03	34.09(+0.35)
reranking+rnn-unnormalized	38.61	38.28	37.74	34.84	24.88	33.94(+0.20)

Table 1. Results for Chinese-English experiments, without (baseline) and with neural LMs for reranking and integrated decoding.

[1] http://nlg.isi.edu/software/nplm/
[2] http://www.fit.vutbr.cz/mikolov/rnnlm/

setting	mt03	mt04	mt05	mt6	mt08	average
baseline	27	50	29	35	25	33.2
decoding+ff-normalized	3355	6563	5352	5691	4816	5155.4(×155.3)
decoding+ff-unnormalized	224	476	251	318	239	301.6(×9.1)
reranking+rnnlm-normalized	153	346	230	273	210	242.4
reranking+rnnlm-unnormalized	59	111	66	85	66	77.4

Table 2. Running time in minutes for MT Chinese-English Evaluation datasets.

In order to manifest the difference between normalized and unnormalized NPLMs, we run SMT experiments under both types of NPLMs, together with baseline setting that only uses two n-gram LMs. The detailed results could be found in table 1. We see that both re-ranking improves around 0.4 BLEU, while direct decoding can get better results up to over 0.85 BLEU. Besides, normalized NPLMs can achieve only slightly better performance(less than 0.05 BLEU on average) on re-ranking task, but un-normalized NLPMs outperform on decoding(also less than 0.05 BLEU on average). Obviously, the difference is so tiny that we believe we can use unnormalized NPLM score. Besides, we also collected the decoding time on different datasets presented on table 2. We observed that decoding with normalized NPLM is on average almost 155 times slower than baseline, and 17.1 times slower than with un-normalized NPLMs. The performance difference between normalized and unnormalized NPLMs for SMT may be the context score is a constant. In order to further verify this conclusion, we check the distribution of the context 4-grams for 1000-best output of MT03 using NPLM trained on Gigaword Xinhua portion, with the histogram plotted in figure 3. We could find that it obeys sharp normal distribution with μ -0.0717 and σ 0.3106, which means that the unnormalized NPLM score could be approximately equal to normalized score. Finally we could draw conclusion based on our experiments that unnormalized feedforward NPLM is sufficient for SMT, which is also consistent with the finding of [15].

Although in Section 2 we know that complexity is already reduced to $O(\sqrt{|V|})$ with decomposition, we want to see whether normalization term can be simply thrown away. As RNNLM cannot be incorporated for direct decoding for hierarchical phrase-based system, we check this idea with reranking only. Similarly, in table 2 reranking with unormalized RNNLMs is three times faster than normalized, but with a heavy loss on BLEU performance(see in table 1). For feedforward NPLM, the normalized term is summed over whole vocabulary, while RNNLM sums over only subset of vocabulary for RNNLM. The histogram of normalized term can be found in figure 4.

6 Incorporating word clusters for NPLMs

As we know the vocabulary size on large training set is usually quite big, e.g., the vocabulary size is 823.6k for Xinhua portion of Gigaword and 286.3k for

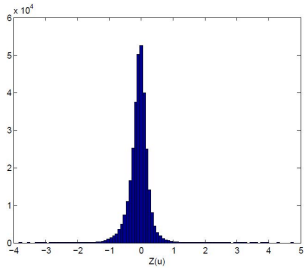

Fig. 3. Histogram of context normalized score of 1000-best of mt03 from feedforward NPLM trained on gigaword.

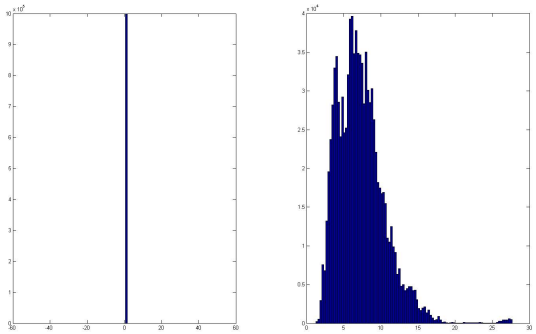

Fig. 4. Histogram of context normalized score of 1000-best of mt03 from RNNLM trained on gigaword, left is classes and right for words.

target side of training data[3]. As a result, it is impractical to train NPLMs on whole vocabulary. Mentioned in section 4, the size was set 100k by default. We want to verify the effect of vocabulary size on BLEU performance. Without loss of generality, we trained both NPLMs on vocabulary size under 12.5k, 25k, 50k, 100k and 200k on both Gigaword Xinhua and target side of training data. The detailed BLEU results are shown in table 4. We can see that performance steadily improves as vocabulary size increases, while it drops when size turns from 100k to 200k. This means that larger vocabulary size does not mean better MT performance, as NPLMs cannot get precise estimation for low-frequency words. In contrast, things are different on n-gram LMs. In details, we trained two n-gram LMs with vocabulary 100k and got BLEU results of both tuning and test sets, which is shown on second row of table 4. We find that what is opposite to NPLMs, full-vocabulary n-gram LMs brings better performance than restricted vocabulary for SMT. Such results show that NPLM cannot obtain good estimate for low-frequency words.

As mentioned in section 4, all out-of-vocabulary words(OOVs) are replaced by some special like <unk>. Therefore there is no difference between two words A and B given same context if they both belong to OOVs, which means the missing of lot of information. Inspired by [26], who improved SMT with word clusters for both n-gram LMs and TMs, we also adopted word classes trained with mkcls[4] for OOVs. In details, each word not in word list is replaced with unk+word-class-label, while no changes for word within the list. With Fixing the vocabulary size as 100k, we trained different clusters-augmented NPLMs with numbers ranged from 250, 500, 1000 to 2000. The final MT results which are illustrated in table 5 show of NPLM augmented with word clusters can

[3] When counting vocabulary size, all digits are converted to zero for fair comparison
[4] code.google.com/p/giza-pp

setting	mt03(dev)	mt04	mt05	mt06	mt08	test average
Baseline	38.17	38.43	37.70	34.25	24.58	33.74
Decoding(ngram-100k)	37.29	38.08	37.09	33.69	24.40	33.31(-0.43)
Decoding+ff-12.5k	38.68	39.02	38.24	34.60	24.71	34.14(+0.40)
Decoding+ff-25k	38.95	39.17	38.75	34.65	24.73	34.33(+0.59)
Decoding+ff-50k	39.01	39.18	38.73	34.68	25.04	34.41(+0.67)
Decoding+ff-100k	39.14	39.49	38.80	34.87	25.31	34.62(+0.88)
Decoding+ff-200k	39.08	39.37	38.85	34.60	25.10	34.48(+0.74)

Table 3. Results for Chinese-English experiments, without neural LM (baseline) and with neural LM for reranking and integrated decoding. For baseline, n-gram LMs are trained with full vocabulary.

Model	Cutoff	
	Gigaword	NIST
Decoding+ff-12.5k	1043	313
Decoding+ff-25k	279	67
Decoding+ff-50k	75	14
Decoding+ff-100k	17	3
Decoding+ff-200k	6	1

Table 4. Results for Chinese-English experiments, without neural LM (baseline) and with neural LM for reranking and integrated decoding. For baseline, n-gram LMs are trained with full vocabulary.

further achieve improvement than decoding with normal NPLMs by nearly 0.3 BLEU when cluster number is larger than 500. The biggest improvement on test dataset(MT05) could reach as high as 1.5 BLEU, which demonstrates that direct decoding with NPLMs plus word clusters achieves higher performance.

setting	mt03(dev)	mt04	mt05	mt06	mt08	test average
Baseline	38.17	38.43	37.70	34.25	24.58	33.74
Decoding+ff-100k	39.14	39.49	38.80	34.87	25.31	34.62(+0.88)
Decoding+ff-100k+c100	39.06	39.31	38.63	35.00	25.30	34.56(+0.82)
Decoding+ff-100k+c500	39.40	39.53	38.78	35.60	25.65	34.89(+1.15)
Decoding+ff-100k+c1k	39.40	39.83	39.20	35.00	25.20	34.81(+1.07)
Decoding+ff-100k+c2k	39.17	39.70	39.05	35.10	25.56	34.86(+1.12)

Table 5. Results for Chinese-English experiments, without neural LM (baseline) and with neural LM for reranking and integrated decoding.

7 Comparison of different types of LMs

As direct decoding with NPLMs can bring significant improvements, one important issue is that whether the NPLMs can replace conventional n-gram language model, which was concerned in[20]. Actually, such comparison has been conducted in terms of perplexity, one latest results could be found in [24]. In following sections, we will discuss this question throughout detailed experiments:

First of all, we tried direct decoding with n-gram or NPLM only. As mentioned in section 3, we only compare feedforward NPLM and n-gram. Table 6 shows the performance of decoding only with n-gram LM and NPLM respectively. Overall speaking, decoding with nplm only is averaged 1.3 BLEU lower than n-gram lms, while feedforward NPLMs augmented with word clusters can achieve better performance, with only about 1.0 BLEU point lower on average. Besides, compared with decoding under n-gram LMs vocabulary size 100k, such gap narrows to 0.9 BLEU and 0.6 BLEU separately. Here we can see that feedforward NPLMs still cannot replace n-gram LMs.

setting	mt03(dev)	mt04	mt05	mt06	mt08	test average
Baseline	38.17	38.43	37.70	34.25	24.58	33.74
Decoding(ngram-100k)	37.29	38.08	37.09	33.69	24.40	33.31(-0.43)
decoding+ff-100k	36.76	37.29	35.70	32.85	23.81	32.41(-1.33)
decoding+ff-100k+c1k	37.15	37.17	36.02	33.35	24.33	32.71(-1.03)

Table 6. Results for Chinese-English experiments, with only ngrams and neural LMs seperately.

In order to further prove this observation, we experimented on all n-gram, feedforward and RNN LMs on a re-ranking task. We first removed language model features of 1000-best outputs from the baseline system. Then different combinations of LMs were appended to output results for re-ranking. Detailed results can be seen on table 7. We may see that when adding one type LM features, similar performance was achieved on feedforward and ngram LMs, but much lower on RNNLM. Similar result can be observed when combing two LMs. Finally the best results can be achieved using all LM features, with a 0.5 Bleu improvement against baseline.

setting	mt03(dev)	mt04	mt05	mt06	mt08	test average
hiero-baseline	38.17	38.43	37.70	34.25	24.57	33.74
reranking-no-lm	36.00	35.99	35.11	32.29	23.03	31.61(-1.93)
reranking+ngrams	38.29	38.10	37.58	34.53	24.71	33.73(-0.01)
reranking+ff	38.54	38.16	37.38	34.79	24.62	33.74 (+0.00)
reranking+rnnlm	38.00	37.99	37.22	34.34	24.42	33.49 (-0.25)
reranking+rnnlm+ff	38.73	38.53	37.79	34.93	24.85	34.03 (+0.29)
reranking+ngrams+ff	38.84	38.62	37.98	35.00	25.08	34.17 (+0.43)
reranking+ngrams+rnnlm	38.69	38.46	37.85	35.04	25.03	34.09(+0.35)
reranking+ngram+rnnlm+ff	39.13	38.43	37.95	35.23	25.30	34.23(+0.49)

Table 7. Results for Chinese-English experiments, with only ngrams and neural LMs seperately.

7.1 Comparison of LMs

In this section, we investigate the possible reason for the difference of performance three types LMs. We focus on the power of models in terms of parameter space, which is listed in table 8 given same training corpus and vocabulary size. All probability information is hard-encoded in n-gram LMs[5]. In contrast, words are mapped to low-dimensional representations via both input and output embeddings. For RNNLM, the context is composed only one previous word and one hidden vector that contains all previous information(represented as one 150×150 matrix). Meanwhile, in addition to both input and output embeddings, feedforward NPLM also comprises two hidden layers(one 600×750 matrix and one 750×150 matrix). We believe the difference of parameters may cause the low performance of NPLMs on SMT.

8 Related Work

Neural network language models have attracted widespread attentions in recent years due to its property to overcome curse of dimension through learning a

[5] Besides, bow information is also stored for back-off

Layer	#Parameter
1-gram	100k
2-gram	5.56m
3-gram	8.87m
4-gram	12.58m
5-gram	13.76m

n-gram

Layer	#Parameter
Input Embeedings	100k×150
Input → Hidden 1	600×750
Hidden 1 → Hidden 2	750×150
Output Embeddings	100k×150

Feedforward

Layer	#Parameter
Input Embeedings	100k×150
Prev Hidden → Hidden	150×150
Output Embeddings	100k×150
4-gram direct connections	2m

RNN

Model	n-gram	feedforward	RNN
Size	1.62G	116.6M	114.6M+7.26M

Model Size

Table 8. *Number of parameters and storage size for three types of LMs trained on gigaword xinhua portion, with vocabulary size 100k. The order for n-gram and feedforward LMs is 5. All parameter settings are illustrated in section 4.*

low-dimensional distributed word representation. Currently there are two types of NPLMs, i.e., feedforward [2] and RNN [12].

We notice that there are some work that tried to apply NPLMs to MT. Schwenk et al. [21, 22] applied feedforward and while Auli et al. [1] and Mikolov [10] investigated RNNLMs. Furthermore Le et al. [9] compared both NPLMs for MT. However, their usage in MT has largely been limited to reranking k-best lists for MT tasks with restricted vocabularies, which could not investigate the role of NPLM for MT in details. Niehues et al. [17] integrated a RBM-based LM directly into a decoder for the first time. The work of Vaswani et al [24] is most related to our work, as they adopted NPLMs for direct decoding on a large-scale MT task, while we try to cover questions that are not covered in their work and further improve the performance.

The problem of normalized NPLM score was first discussed for log-bilinear model [15]. Later on both Niehues et al. [17] and Vaswani et al. [24] adopted unnormalized score for direct decoding, however they did not further investigate normalized score for MT and for other NPLMs(e.g., RNNLM), which were both discussed in this paper.

Word clustering has been used in language modeling for a long period [3]. Weubker et al. [26] improved MT with class-based n-gram LMs. Besides, Wu et al. [25] demonstrated that factored RNNLM with knowledge like POS and stem could outperform conventional RNNLM. However, as words can have more than one POS tag, which need to be inferred for the words in translation rules. Instead we used word cluster for truncated words to train NPLMs and also get further improvements.

Recent progress has shown the power of different NPLMs. We notice that Sundermeyer et al. [23] compared feedforward with RNN LMs on speech recog-

nition task. However there is no systematic comparison of neural models against n-gram LM, which we believe is an important question for MT research.

9 Conclusion

In this work, we discuss several questions that exist in the application of NPLMs for SMT system. First of all, through detailed experiments we show that un-normalized feedforward NPLM is equivalent as normalized for decoding and reranking, while removing softmax can reduce the decoding time to 1/10. However, unnormalized RNNLM cannot replace normalized ones as it uses decomposition of probability. We also show that simply increasing vocabulary size of feedforward NPLMs cannot improve MT performance. Instead, replacing OOVs with cluster of original word can make better estimation of NPLMs and bring another 0.2 BLEU improvement for direct MT decoding. Finally, experiments also show that for MT both types of NPLMs might not simply replace conventional n-gram LM. We hope our findings can benefit the research on NPLMs in SMT.

In future, we will investigate following directions: first of all, we will conduct more experiments with direct decoding with RNNLMs for phrase-based MT system [1], in which way the power of RNNLM can be directly tested; secondly, we will compare the NCE training used in NPLM with self-normalization in neural network joint model(NNJM) [7]; we also plan to incorporate linguistic and domain information into the neural models.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (No. 61223003), Specialized Research Fund for the Doctoral Program of Higher Education of China(No. 20110091110003) and Graduate Research and Innovation Projects in Jiangsu Province(No. CXZZ12_0058). Shujian Huang is the corresponding author.

References

1. Auli, M., Galley, M., Quirk, C., Zweig, G.: Joint language and translation modeling with recurrent neural networks. In: Proceedings of the 2013 Conference on EMNLP. pp. 1044–1054 (2013)
2. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. Journal of Machine Learning Research (2003)
3. Brown, P.F., Desouza, P.V., Mercer, R.L., Pietra, V.J.D., Lai, J.C.: Class-based n-gram models of natural language. Computational linguistics 18(4), 467–479 (1992)
4. Chen, S.F., Goodman, J.: An empirical study of smoothing techniques for language modeling. Tech. Rep. TR-10-98, Harvard University Center for Research in Computing Technology (1998)

5. Chiang, D.: Hierarchical phrase-based translation. Computational Linguistics 33(2), 201–228 (2007)
6. Clark, J.H., Dyer, C., Lavie, A., Smith, N.A.: Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 176–181. Association for Computational Linguistics, Portland, Oregon, USA (June 2011), http://www.aclweb.org/anthology/P11-2031
7. Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., Makhoul, J.: Fast and robust neural network joint models for statistical machine translation. In: Proceedings of the ACL. Association for Computational Linguistics, Baltimore, Maryland (June 2014)
8. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: Proceedings of AISTATS (2010)
9. Le, H.S., Allauzen, A., Yvon, F.: Measuring the influence of long range dependencies with neural network language models. In: Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT. Montréal, Canada (2012)
10. Mikolov, T.: Statistical Language Models Based on Neural Networks. Ph.D. thesis, Brno University of Technology (2012)
11. Mikolov, T., Deoras, A., Kombrink, S., Burget, L., Černocký, J.H.: Empirical evaluation and combination of advanced language modeling techniques. In: Proceedings of INTERSPEECH. pp. 605–608 (2011)
12. Mikolov, T., Karafiát, M., Burget, L., Černocký, J.H., Khudanpur, S.: Recurrent neural network based language model. In: Proceedings of INTERSPEECH (2010)
13. Mnih, A., Hinton, G.: Three new graphical models for statistical language modelling. In: Proceedings of ICML (2007)
14. Mnih, A., Hinton, G.: A scalable hierarchical distributed language model. In: NIPS (2009)
15. Mnih, A., Teh, Y.W.: A fast and simple algorithm for training neural probabilistic language models. In: Proceedings of the 29th ICML. pp. 1751–1758 (2012)
16. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of ICML. pp. 807–814 (2010)
17. Niehues, J., Waibel, A.: Continuous space language models using Restricted Boltzmann Machines. In: Proceedings of IWSLT (2012)
18. Och, F.J.: Minimum error rate training in statistical machine translation. In: Proceedings of ACL. pp. 160–167 (2003)
19. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics 29(1), 19–51 (2003)
20. Ramabhadran, B., Khudanpur, S., Arisoy, E. (eds.): Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT. Montréal, Canada (June 2012)
21. Schwenk, H.: Continuous space language models. Computer Speech and Language 21, 492–518 (2007)
22. Schwenk, H.: Continuous-space language models for statistical machine translation. Prague Bulletin of Mathematical Linguistics 93, 137–146 (2010)
23. Sundermeyer, M., Oparin, I., Gauvain, J.L., Freiberg, B., Schluter1, R., Ney, H.: Comparison of feedforward and recurrent neural network language models. In: Proceedings of ICASSP (2013)

24. Vaswani, A., Zhao, Y., Fossum, V., Chiang, D.: Decoding with large-scale neural language models improves translation. In: Proceedings of the 2013 Conference on EMNLP. pp. 1387–1392
25. Wu, Y., Lu, X., Yamamoto, H., Matsuda, S., Hori, C., Kashioka, H.: Factored language model based on recurrent neural network. In: Proceedings of COLING 2012. pp. 2835–2850. Mumbai, India (December 2012)
26. Wuebker, J., Peitz, S., Rietig, F., Ney, H.: Improving statistical machine translation with word class models. In: Proceedings of EMNLP. pp. 1377–1381 (2013)