SEMANTIC LANGUAGE MODELS FOR AUTOMATIC SPEECH RECOGNITION

Ali Orkan Bayer and Giuseppe Riccardi

Signals and Interactive Systems Lab - University of Trento, Italy
{bayer, riccardi}@disi.unitn.it

ABSTRACT

We are interested in the problem of semantics-aware training of language models (LMs) for Automatic Speech Recognition (ASR). Traditional language modeling research have ignored semantic constraints and focused on limited size histories of words. Semantic structures may provide information to capture lexically realized long-range dependencies as well as the linguistic scene of a speech utterance. In this paper, we present a novel semantic LM (SELM) that is based on the theory of frame semantics. Frame semantics analyzes meaning of words by considering their role in the semantic frames they occur and by considering their syntactic properties. We show that by integrating semantic frames and target words into recurrent neural network LMs we can gain significant improvements in perplexity and word error rates. We have evaluated the semantic LM on the publicly available ASR baselines on the Wall Street Journal (WSJ) corpus. SELMs achieve 50% and 64% relative reduction in perplexity compared to n-gram models by using frames and target words respectively. In addition, 12% and 7% relative improvements in word error rates are achieved by SELMs on the Nov'92 and Nov'93 test sets with respect to the baseline tri-gram LM.

***Index Terms*—** Language Modeling, Recurrent Neural Networks, Frame Semantics, Semantic Language Models

1. INTRODUCTION

Language models (LMs) constrain the search space of automatic speech recognition (ASR) systems by estimating probabilities for possible sequences of words. The most widely used LMs are n-grams. However, as discussed in [1] n-grams consider almost no linguistic information. One of the problems related with this is the locality problem [2], i.e. n-grams are based on fixed length of histories and they fail to capture long-range dependencies. As stated by [2] long-range dependencies can be handled in LMs either by syntactic information or semantic information. In this paper, we show how linguistically aware LMs can be built by using semantic information that is based on the theory of frame semantics.

Long-distance modeling has been addressed by using trigger based LMs [3]. In this case, the probability estimates of word sequences are modified by the co-occurrence of their triggers. Linguistic information was encoded into LMs by [4] where the next-word probabilities are estimated using the syntactic structures generated by the parser. Although the meaning of speech utterance is of primary importance in human-machine spoken interaction, semantics for LMs has received little attention. This is due to the limitations of a) general theory of semantics suitable for language modeling b) accurate and fast semantic parser. The advantages of semantic features in LMs are a) early semantic constraint processing for ASR and b) extendible approach multimodal and situated language modeling and understanding. The most notable use of semantics in LM is the latent semantic analysis work in [2] and the *recognition for understanding* LM training in [5]. Recent improvements in LMs came from the introduction of neural network LMs (NNLMs) [6]. NNLMs project the discrete word space onto a continuous space, in this way probability distributions of words can be estimated effectively [7]. Feedforward NNLMs use a fixed history, on the other hand, recurrent NNLMs (RNNLMs) [8] use recurrent connections and model a short term memory that represents the state of the network. In [9], a cache NNLM is presented for spoken language understanding tasks, which uses an additional cache layer. Context dependent RNNLMs are presented in [10], which use an additional layer similar to the cache NNLM. This additional layer models the long-span context.

In this paper we exploit the theory of Frame Semantics to train language models for ASR. Frame semantics is an area of lexical semantics where the meaning of words are analyzed in the frames that they occur [11] — the linguistic scene. In this paper, we propose and evaluate the automatic training algorithms for SELMs. We demonstrate how frame semantics can be used to improve the performance of LMs. We use the frames evoked and the frame-evoking predicates as semantic features. For this purpose, context dependent RNNLMs are used with semantic features as context. We evaluate the performance of semantic LMs over perplexity on Wall Street Journal part of Penn-Treebank and over word error rate (WER) on the Wall Street Journal (WSJ) speech corpus.

This paper is organized as follows. Section 2 describes the

The research leading to these results has received funding from the European Union – Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 610916 – SENSEI.

semantic feature extraction step. Section 3 gives the details of SELMs. Section 4 presents the perplexity results on Penn-Treebank. Section 5 explains how semantic LMs can be used for re-scoring N-best ASR hypotheses, and presents the WER results on WSJ corpus. Finally, section 6 gives concluding remarks.

2. SEMANTIC FEATURE EXTRACTION

Frame semantics is a theory of lexical meaning where word meanings are described in the context of semantic frames, which are evoked by linguistic forms. FrameNet is a project which analyzes semantic frames and word meanings in relation to these frames. In this framework, words can evoke semantic *frames* depending on their senses. Frame evoking words are called *target words or targets*. On the other hand, they can participate in frames to complete the meaning, then they are called *frame elements*. In this paper, we use *frames* and *targets* for semantic feature extraction [11].

Frame semantic parsing is the process of extracting the semantic information that corresponds to target words, frames evoked, and frame elements. We have used the open-source frame-semantic parser, SEMAFOR [12], for extracting semantic features. SEMAFOR performs semantic parsing in three steps. The first step is the rule-based target identification step, in which the frame evoking predicates, i.e. the targets, are recognized. The next step identifies the frames evoked by these predicates by using a statistical model. The frame elements are filled as the final step by using another statistical model. SEMAFOR relies on the output of a dependency parser. The reader should refer to [12] for a detailed description of SEMAFOR semantic parser.

The SELM uses two different semantic features. Each utterance is passed through SEMAFOR frame-semantic parser, and *frames* and *targets* are extracted. Then a semantic feature vector is constructed based on this information. When *frames* are used, we set the index of the evoked frames to 1 and the rest to 0 to create the feature vector. When creating feature vectors we do not consider the frequencies, therefore even if a *frame* is evoked more than once its index is set to 1; as shown in Figure 1. We create the feature vectors for *targets* similarly. These feature vectors are used as the *semantic context* for that utterance in the SELM. Therefore, for each utterance the *semantic context* is fixed.

3. SELM STRUCTURE

The SELMs presented in this paper, use RNNLMs as the main building block. RNNLMs have been introduced in [8] and are shown to reduce perplexity and WER significantly. We have used a similar structure to context dependent RNNLMs that are introduced in [10]. RNNLMs employ recurrent connections to represent the state of the network through time. This state, together with the current word, constitute the history that the probability of the next word is estimated on.

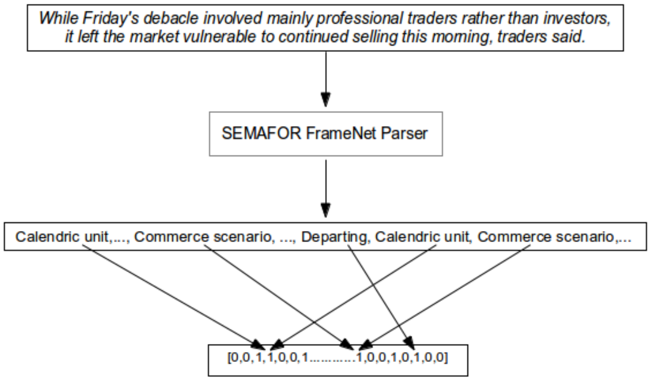
While Friday's debacle involved mainly professional traders rather than investors, it left the market vulnerable to continued selling this moming, traders said.
SEMAFOR FrameNet Parser
Calendric unit,..., Commerce scenario, ..., Departing, Calendric unit, Commerce scenario,...
[0,0,1,1,0,0,1............1,0,0,1,0,1,0,0]

Fig. 1. Semantic feature extraction. An utterance is fed to the SEMAFOR frame semantics parser as input, the parser outputs the frames evoked for that utterance. The feature vector is created by using the output of the semantic parser.

The main complexity of an RNNLM depends on the size of the vocabulary. One of the solutions to this problem is to use class-based RNNLMs that are presented in [13]. It is also possible to train a maximum-entropy model that uses n-gram features together with an RNNLM, which is shown to improve WER [14]. RNNLMs that use a maximum-entropy model with n-gram features are referred to as RNNME models [14, 10]. In this paper, we have used the open-source RNNLM toolkit [15]. The toolkit already employs the class-based approach and the maximum-entropy model training. We have modified the toolkit and added a context layer which would be used as the semantic context.

RNNLMs are composed of an input layer which has the size of the vocabulary, a hidden layer which has recurrent connections to the recurrent layer that represent the hidden state of the network, and an output layer. The output layer, in the class-based implementation, estimates the word probabilities by factorizing them into class probabilities and class-membership probabilities. The input is encoded as 1-of-n encoding. In SELMs an additional context layer is used to represent the semantic context for the current utterance. The SELM is depicted in Figure 2. The SELM is trained by using the backpropagation through time (BPTT) algorithm, where the network is unfolded for N time steps back and the weights are updated by using the standard backpropagation algorithm [8]. The maximum-entropy model that uses n-gram features are implemented as direct connections between n-gram histories and the output layer (which are not shown in the figure), n-gram histories are further implemented by hashing. The details of this implementation can be found in [14].

3.1. Word prediction with SELM: an example

The intuitive idea behind SELMs is that the linguistic scene that is constructed by semantic information would help to predict relevant words better. We show how this works practically on the following sentence from Penn-Treebank:
“While Friday’s debacle involved mainly professional traders

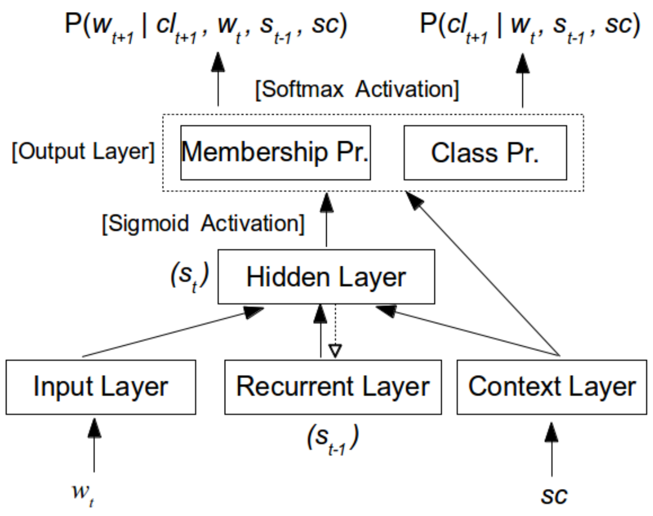

Fig. 2. The SELM structure that is based on the class-based RNNLM structure. The network takes the current word w_t and the semantic context sc for the current utterance as input. In addition the previous hidden state is copied into the recurrent layer s_{t-1}. The output layer estimates the probability for the next word w_{t+1} factorized into class probabilities and class-membership probabilities (cl_{t+1} denotes the recognized class for the next word). Therefore these probabilities are conditioned on the current word w_t, the previous state s_{t-1} and the semantic context sc. The direct connections from n-gram histories to output layers are not shown.

rather than investors, it left the ***market*** *vulnerable to continued selling this morning, traders said.*"

When passed through the semantic parser, the parser recognizes the following frames:
Calendric unit, Catastrophe, Participation, People by vocation, ***Commerce scenario****, Departing, Being at risk, Activity ongoing,* ***Commerce sell****, Calendric unit,* ***Commerce scenario****, Statement*

We demonstrate how the probability for the word *market* (shown in bold), which is not a target, is estimated by some of the LMs that are presented in Section 4. In addition, we replace the word, *market*, with an irrelevant word, *computer*, in the same sentence. The probability estimates for *market* and *computer* in the same context by an n-gram LM, a standard RNNME model, and a SELM are given in Table 1.

Table 1. Probability estimates of the word *market* and the substituted word *computer* given the history (h). For the 5-gram LM, h is the preceding words; and for the RNNME the preceding word and the hidden state. h, for the SELM is the preceding word, the hidden state and the semantic frames.

Model	**P(*market* | h)**	**P(*computer* | h)**
Kneser-Ney 5-gram	4.2×10^{-3}	8.2×10^{-4}
RNNME	6.9×10^{-3}	1.8×10^{-3}
SELM on Frames	1.2×10^{-2}	1.9×10^{-5}

As can be seen in Table 1, SELM that uses *frames* as semantic context, estimates a higher probability for the relevant word, *market*. In addition, it assigns a lower probability to an irrelevant word, *computer*. We believe that, the linguistic scene built by semantic frames *Commerce scenario* and *Commerce sell* are effective in this better estimation.

4. PENN-TREEBANK EXPERIMENTS

In this section we present the perplexity results on the publicly available Penn-Treebank part of the WSJ corpus. The experiments presented here are performed on the same data and with the same preprocessing steps (with the same training/testing partitions and the same vocabulary) given in [16, 17, 10].

The preprocessing steps involve representing numerical values with the special token *"N"* and limiting the vocabulary to the most frequent 10K tokens, all other tokens are mapped to an *unknown* token. We have used the following split. Sections 0-20 are used for training, sections 21-22 are used as the development set, and sections 23-24 are used as the evaluation set. The number of tokens are 930K, 74K, and 82K for training, development and evaluation sets respectively.

The semantic features are extracted on the raw data that is not preprocessed. Therefore, the raw data is fed to the semantic parser and semantic features are extracted over the frames and the targets by using the semantic feature extraction step. For the Penn-Treebank we have 819 distinct frames and 11271 distinct targets in the training set.

We have trained a 5-gram Kneser-Ney LM with singleton cut-offs (KN5), a 4-gram feed-forward NNLM that has 160 nodes in the hidden layer and uses 200 word classes that are assigned with respect to their frequencies (FF4), and a RNNME model that uses the same clustering of words, it has 150 nodes in the hidden layer and uses 4-gram features for the maximum entropy model with a size of 10^9 connections (RNNME). The NNLMs are optimized over the perplexities on the development set for their size of hidden layers and their random initializations.

The SELMs we have built are RNNME models with semantic context. The SELMs are trained with the semantic context over frames ("SELM on Frames") and over targets ("SELM on Targets"). All SELMs use 200 word classes that are same with the previous NNLMs to reduce the computational complexity of training, and they use 4-gram features for the maximum entropy model with a size of 10^9 connections. They have 200 nodes in the hidden layer. In addition to these models, we have trained SELMs by using the most frequent frames and targets that cover the 80% of the training data. This reduces the size of distinct frames to 181 and distinct targets to 1386, therefore this also reduces the computational complexity of the training procedure. The perplexities of all LMs are presented in Table 2.

We have achieved 50% and 64% relative reduction in perplexity with respect to the Kneser-Ney 5-gram LM by using frames and targets as semantic context. In addition, we have

Table 2. Perplexity results on Penn-Treebank part of the Wall Street Journal corpus. SELMs achieve 50% and 64% relative reduction in perplexity with respect to Kneser-Ney 5-gram model when frames and targets are used as semantic context respectively.

Model	**Dev PPL**	**Test PPL**
KN5	148.0	141.2
FF4	165.9	156.3
RNNME	133.6	127.9
SELM on Frames	**73.7**	**70.3**
SELM on 80% Frames	84.6	81.4
SELM on Targets	**53.8**	**51.1**
SELM on 80% Targets	63.3	60.5

achieved 5% and 31% relative reduction in perplexity with respect to the lowest reported results in [10]. Restricting the frame size and targets to a coverage of 80% also achieves a good reduction in perplexity. Therefore, for the re-scoring ASR experiments on a larger corpus, we have used the frames and targets with 80% coverage to reduce the training complexity.

5. WALL STREET JOURNAL EXPERIMENTS

In this section we present the results on N-best re-scoring experiments on the WSJ speech recognition task. All of the experiments presented in this section are performed by using the publicly available WSJ0/WSJ1 (DARPA November'92 and November'93 Benchmark) sets. The acoustic models are trained on the WSJ0/WSJ1 training utterances also known as SI-284. All the development data under WSJ1 for speaker independent 20k vocabulary is used as the development set ("Dev 93" - 503 utterances). The evaluation is done on the November 92 CSR Speaker independent 20k NVP test set ("Test 92" - 333 utterances) and on the November 93 CSR HUB 1 test set ("Test 93" - 213 utterances).

5.1. ASR baseline

The baseline ASR system is built by using the Kaldi speech recognition toolkit [18]. This system generates the N-best lists that are used for re-scoring. The vocabulary is set to 20K by using the 20K open vocabulary word list for non-verbalized punctuation that is available in WSJ0/WSJ1 corpus. The language model that the baseline system uses is the baseline tri-gram backoff model for 20K open vocabulary for non-verbalized punctuation that is also available in WSJ0/WSJ1 corpus.

The acoustic models are trained over the SI-284 data by using the publicly available Kaldi recipe with the following settings. MFCCs features are extracted and spliced in time with a context window of $[-3, +3]$. Linear discriminant analysis (LDA) and maximum likelihood linear transform (MLLT) are applied. Triphone Gaussian mixture models are trained over these features.

The ASR baseline performs weighted finite state decoding. We have extracted 100-best lists for each development and evaluation set. The performance of ASR baseline is given in Table 3.

Table 3. The WER performance of the ASR baseline system on Dev 93, Test 92, and Test 93 sets.

	Dev 93	**Test 92**	**Test 93**
ASR 1-best	15.3%	10.2%	14.0%
Oracle on 100-best	8.3%	5.1%	7.3%

5.2. Re-scoring experiments

Re-scoring experiments are performed on the 100-best lists that are generated by the baseline ASR system. These 100-best lists are re-scored by using the SELMs. In addition, we have trained n-gram LMs and NNLMs to better compare the SELMs with. All LMs are trained over the whole WSJ 87, 88, and 89 data with the vocabulary that is limited to the 20K open vocabulary for non-verbalized punctuation. The LMs for comparison include a Kneser-Ney 5-gram model with singleton cut-offs (KN5), a 4-gram feed-forward NNLM that has 240 nodes in the hidden layer and with a projection layer of size 64 (FF4), and a RNNME model that has 20 nodes in the hidden layer and uses a maximum-entropy model that has 4-gram features with 10^9 connections (RNNME). The KN5 model is built on words without any classes. However, to reduce the training time FF4 and RNNME are trained by using 200 word classes that are constructed with respect to the frequencies of the words. The NNLMs are tuned to the lowest WER on the development set by using different sizes of hidden layers and with different random initializations. The performances of these models are given in Table 4.

Table 4. The WER performance of the 5-gram LM, the feed-forward LM, and the RNNME model.

Model	**Dev 93**	**Test 92**	**Test 93**
KN5	14.5%	9.6%	13.4%
FF4	14.6%	9.6%	13.9%
RNNME	14.2%	9.3%	13.1%

We have trained SELMs that use frames and targets as semantic context separately. The SELMs are also trained on the same data with the same vocabulary setting. They use the same word classes that are used by the previous NNLMs. The semantic features for each utterance in the training data are extracted by feeding them to the semantic parser SEMAFOR. The training data has 841 distinct frames and 17736 distinct targets. We have limited the number of frames and targets to the most frequent ones that cover the 80% of the training data, which results in 184 distinct frames and 1182 distinct targets. We have trained the SELMs by using the BPTT algorithm on the training data. We have used the reference transcription and reference semantic context of the Dev 93 set as the validation set for early stopping to avoid overfitting.

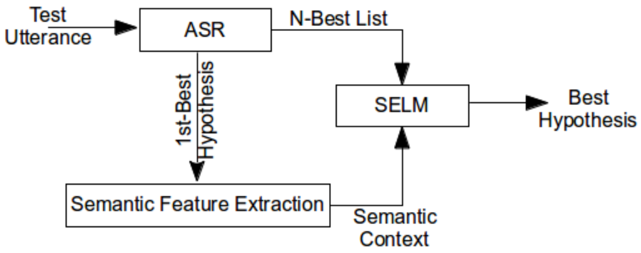

Fig. 3. The flow of re-scoring experiments. The test utterance is passed through the baseline ASR. The ASR 1-best hypothesis is given to the semantic feature extraction module, which extracts the semantic context for that utterance. The N-best list is re-scored by using the SELM with the semantic context for that utterance.

The re-scoring experiments by using the semantic LMs are conducted by using the following setting. The semantic context for the utterance that will be re-scored can be extracted either from the reference transcription, oracle hypothesis, or the ASR hypothesis. Naturally, the experiments that use the semantic context of the ASR hypothesis will reflect the real performance. The others can be used to see the upper bound of the performance. Therefore, we refer to the output of the semantic parser as follows. The output of the semantic parser (frames and targets) on the reference transcription are referred to as *reference frames and reference targets*. The output of the parser on the ASR output are referred to as *ASR frames and ASR targets*. Finally, the output on the oracle hypotheses are referred to as *oracle frames and oracle targets*. We present the results on reference frames/targets and on oracle frames/targets to present an upper bound on the performance of the SELMs, the actual performance is given by the ASR frames/targets. The re-scoring procedure for ASR frames/targets is depicted in Figure 3.

Table 5. The WER performance of the SELMs. The bold WERs (ASR Frames and ASR Targets) present the actual performance. Results on the reference and oracle frames/targets are given to show an upper bound.

Model	Dev 93	Test 92	Test 93
SELM on Frames			
Reference Frames	13.4%	8.7%	12.3%
Oracle Frames	13.2%	8.7%	12.0%
ASR Frames	**14.5%**	**9.5%**	**13.9%**
SELM on Targets			
Reference Targets	12.9%	8.4%	11.7%
Oracle Targets	12.9%	8.4%	11.6%
ASR Targets	**15.0%**	**10.0%**	**14.4%**

As can be seen in Table 5 when accurate semantic information (reference frames/targets and oracle frames/targets) is used as the semantic context, the SELMs achieve a significant improvement in WER. Target words, since they are more constraining on semantics, give better results. However, actual performance, i.e. when the ASR frames and targets are used, is affected by the noise in the semantic context. We also observe that frames as semantic features are more robust to this noise.

5.3. Making sense of semantic context

The results in Table 5 show the potential performance of SELMs. When SELMs are supplied with accurate semantic context, their performance significantly improves. However, the noise on the ASR frames and targets drops their performance to an unacceptable range. Therefore to improve the actual performance, thus to lower the noise on the semantic context, we have eliminated the frames and targets that have high error rate on the ASR hypothesis. This error is computed on ASR frames and targets with respect to the reference frames and targets. Thus, we have eliminated these frames and targets which have an error rate of 10% on the development set. After elimination, we have ended up with 60 distinct frames and 541 distinct targets. The SELMs are trained from scratch by using this subset of frames and targets and re-scoring experiments are repeated with these new SELMs. The performance of the these models is given in Table 6.

Table 6. Improved WER performance of the SELMs by using low error frames and targets. The results show that by eliminating erroneous frames and targets, we can get significant improvements on WER with ASR frames and targets (given in bold). The SELM on Frames achieve 12% relative improvement on Test 92 evaluation set and 7% relative improvement on Test 93 evaluation set with respect to the ASR baseline.

Model	Dev 93	Test 92	Test 93
SELM on Frames			
Reference Frames	13.6%	8.9%	13.0%
Oracle Frames	13.5%	8.9%	12.8%
ASR Frames	**13.8%**	**9.0%**	**13.0%**
SELM on Targets			
Reference Targets	13.7%	8.9%	13.1%
Oracle Targets	13.7%	8.9%	13.0%
ASR Targets	**13.9%**	**9.5%**	**13.9%**

The results on Table 6 show that eliminating the erroneous frames and targets yields significant improvement with ASR frames and targets. Since this elimination is done on the development set, the development set benefits more, especially on ASR targets. The SELM with ASR frames achieves a relative improvement of 12% and 7% on Test92 and Test 93 evaluation sets with respect to the ASR baseline.

5.4. Model combination

It is possible to obtain more improvement by linearly interpolating NNLMs with n-gram models. We have optimized the weights of linear interpolation over WER on the Dev 93 set. Table 7 presents the combination of two models. The first one is the combination of the 5-gram model (KN5) with the RNNME model (Table 4, RNNME). The second one is the combination of the 5-gram model (KN5) with the SELM on ASR Frames (Table 6, SELM on Frames). We observe that the combination with the SELM gives a better performance than the combination with the RNNME model. The combination with the SELM achieves 14% and 11% relative improvement with respect to the ASR baseline on Test 92 and Test 93 respectively.

Table 7. Linear interpolation of LMs. The combination with the SELM gives a better performance than the combination with the RNNME.

Model	**Dev 93**	**Test 92**	**Test 93**
KN5 + RNNME	13.8%	9.2%	13.0%
KN5 + SELM on ASR Frames	13.4%	8.8%	12.5%

6. CONCLUSION

Semantic information helps to capture the long-span dependencies that linguistic constructions have. This paper presents a novel SELM that is based on the theory of frame semantics. We have constructed SELMs by using context dependent RNNME models. The semantic context is extracted from evoked frames and targets in an utterance. We have achieved significant reductions in perplexity on Penn-Treebank. In addition, by performing re-scoring experiments on WSJ speech recognition corpus, we have obtained significant improvements in WER by using SELMs that use frames as semantic context. We observe that SELMs on frames performs better than standard RNNME models even in model combination with n-gram models.

7. REFERENCES

[1] R. Rosenfeld, “Two decades of statistical language modeling: where do we go from here?,” *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1270–1278, Aug 2000.

[2] J.R. Bellegarda, “Exploiting latent semantic information in statistical language modeling,” *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1279–1296, Aug 2000.

[3] R. Lau, R. Rosenfeld, and S. Roukos, “Trigger-based language models: a maximum entropy approach,” in *Proceedings of ICASSP*, April 1993, vol. 2, pp. 45–48.

[4] C. Chelba and F. Jelinek, “Structured language modeling,” *Computer Speech & Language*, vol. 14, no. 4, pp. 283–332, 2000.

[5] G. Riccardi and A. L. Gorin, “Stochastic language models for speech recognition and understanding,” in *Proceedings of ICSLP*, 1998.

[6] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, “A neural probabilistic language model,” *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2000.

[7] H. Schwenk, “Continuous space language models,” *Computer Speech Language*, vol. 21, no. 3, pp. 492–518, 2007.

[8] T. Mikolov, M. Karafiat, L. Burget, J. Cernock, and S. Khudanpur, “Recurrent neural network based language model,” in *Proceedings of Interspeech*. 2010, pp. 1045–1048, ISCA.

[9] F. Zamora-Martinez, S. Espana-Boquera, J. Castro-Bleda, M., and R. De-Mori, “Cache neural network language models based on long-distance dependencies for a spoken dialog system,” in *Proceedings of ICASSP*. 2012, IEEE.

[10] T. Mikolov and G. Zweig, “Context dependent recurrent neural network language model.,” in *Proceedings of SLT*. 2012, pp. 234–239, IEEE.

[11] Charles J. Fillmore, Christopher R. Johnson, and Miriam R. L. Petruck, “Background to Framenet,” *International Journal of Lexicography*, vol. 16, no. 3, pp. 235–250, Sept. 2003.

[12] D. Das, D. Chen, A. F. T. Martins, N. Schneider, and N. Smith, “Frame-semantic parsing,” *Computational Linguistics*, vol. 40, no. 1, pp. 9–56, 2014.

[13] T. Mikolov, S. Kombrink, L. Burget, J.H. Cernocky, and Sanjeev Khudanpur, “Extensions of recurrent neural network language model,” in *Proceedings of ICASSP*, May 2011, pp. 5528–5531.

[14] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. ernock, “Strategies for training large scale neural network language models,” in *Proceedings of ASRU*. 2011, pp. 196–201, IEEE.

[15] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. ernock, “RNNLM - recurrent neural network language modeling toolkit,” in *Proceedings of ASRU*. 2011, pp. 1–4, IEEE.

[16] A. Emami and F. Jelinek, “Exact training of a neural syntactic language model.,” in *Proceedings of ICASSP*. 2004, pp. I–245–8 vol.1, IEEE.

[17] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernockỳ, “Empirical evaluation and combination of advanced language modeling techniques.,” in *Proceedings of Interspeech*, 2011, pp. 605–608.

[18] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, “The kaldi speech recognition toolkit,” in *Proceedings of ASRU*. 2011, IEEE.