Contextual spoken language understanding using recurrent neural networks

Conference Paper · April 2015

DOI: 10.1109/ICASSP.2015.7178977

CONTEXTUAL SPOKEN LANGUAGE UNDERSTANDING USING RECURRENT NEURAL NETWORKS

Yangyang Shi, Kaisheng Yao, Hu Chen, Yi-Cheng Pan, Mei-Yuh Hwang, Baolin Peng

Microsoft

ABSTRACT

We present a contextual spoken language understanding (contextual SLU) method using Recurrent Neural Networks (RNNs). Previous work has shown that context information, specifically the previously estimated domain assignment, is helpful for domain identification. We further show that other context information such as the previously estimated intent and slot labels are useful for both intent classification and slot filling tasks in SLU. We propose a step-n-gram model to extract sentence-level features from RNNs, which extract sequential features. The step-n-gram model is used together with a stack of Convolution Networks for training domain/intent classification. Our method therefore exploits possible correlations among domain/intent classification and slot filling and incorporates context information from the past predictions of domain/intent and slots. The proposed method obtains new state-of-the-art results on ATIS and improved performances over baseline techniques such as conditional random fields (CRFs) on a large context-sensitive SLU dataset.

Index Terms— Recurrent Neural Networks, Convolution Networks, Spoken Language Understanding

1. INTRODUCTION

A Spoken Language Understanding (SLU) system consists of domain identification, intent classification and slot filling [1]. SLU is a critical component in spoken dialogue systems. SLU usually processes an input query in a sequential way; it firstly identifies the domain of a query, then classifies its intent and lastly extracts semantic slots from the natural language query. For example, a query of “I want to fly from Seattle to Paris,” should be classified as having an intention of ”Departure” in the domain of ”Flight”. The word “Seattle” should be labeled as the departure-city of a trip, and “Paris” as the arrival-city.

SLU has been extensively studied in recent years. Domain identification, intent classification, and slot tagging are usually modeled separately. Therefore, perhaps the most obvious approach to SLU is treating both domain and intent detection as classification problems using Support Vector Machines (SVMs) [2], and using sequence labeling methods such as Conditional Random Field (CRF) [3] for slot tagging task that assigns each word in the query with a slot label.

In a contextual SLU, users interact with the system via a sequence of consecutive natural language queries. Each query has its intra-session history information from previous queries, associated with their previously predicted domains, intents and slot labels. An example of a user session consisting of 5 queries/turns is as follows:

query	domain	intent
T1: what is the weather in Nanjing	weather	check weather
T2: how about Shanghai	weather	check weather
T3: how can I go to Shanghai	places	get route
T4: set an alarm for me	alarm	set alarm
T5: seven o’clock in the morning	alarm	set alarm

In this example, the user starts with weather queries of two cities, then asks for route information of the second city. The user finally asks for setting up an alarm clock with a specific time. It would be ambiguous to SLU if the history information of domain, intent, and slot labels are ignored, especially for queries T2, T3, and T5; the features extracted from current query are insufficient for statistical models to make an accurate judgment. For example, “seven o’clock in the morning” can have different domain/intent interpretations such as ”set alarm” or ”time”. It is shown in [4, 5] that these context information are useful to improve system performance on domain classification.

In recent years, Recurrent Neural Network (RNN) has demonstrated outstanding performance in a variety of natural language processing tasks [6–14]. In common with feed-forward neural networks [15–19], an RNN maintains a representation for each word input as a high-dimensional real-valued vector. In this vector space, similar words tend to be close with each other, and relationships between words are preserved [20]; thus, adjusting the model parameters to increase the objective function for a training example which involves a particular word tends to improve performance for similar words in similar contexts. RNN, however, differs from the Feed-Forward Neural Networks in that RNN has a recurrent connections of hidden layer activities. Therefore, the current hidden layer activities incur memories of the past activities. This may make RNN particularly suitable for sequence labeling tasks such as slot filling, in which RNN has demonstrated outstanding performances on some benchmark tasks [12].

In this paper, we propose a RNN-based SLU approach to joint training of domain identification, intent classification and slot filling for contextual SLU. The RNN in our method performs slot filling tasks. Its features are however jointly trained for both slot filling and domain/intent assignment. A Convolution Neural Network (CNN) [21] is placed on top of features extracted from a step-n-gram model, which extracts sentence-level features from sequential hidden layer activities from RNN. One obvious advantage of the proposed approach is that it can jointly train neural networks parameters. In contrast, conventional methods for SLU trains model domain/intent classification and slot filling separately. Another advantage of the proposed method is that the context information, including domain/intent assignment and slot tagging from the past predictions, are incorporated.

2. RELATED WORK

Due to its simplicity and robustness, Support Vector Machine with linear kernel (linear SVM) [2] has been widely used for domain and intent classification. Recently, Neural Networks and Convolution

We would like to thank Puyang Xu for helpful discussions.

Neural Networks have also been applied to domain classification and intent detection [5,22–25]. A linear SVM may be considered shallow, in comparison with the neural network based models.

Conditional Random Field (CRF) [3] has been extensively used for slot filling. Recently NN-based technologies [22,26,27] and RNN-based methods [12, 28–30] have been successfully used on the task. The RNN-based method achieved the state-of-the-art performance on some datasets such as ATIS [29, 31]. Our work in this paper further advance the state-of-the-art by joint training of slot filling with the task of domain/intent classification.

The joint training is related to multi-tasks learning [26]. Recently, [4, 25] have investigated joint modeling of intent detection and slot filling. This paper further their works by incorporating the contextual information from the predictions of domain/intent classification and slot filling of the previous queries. When slot labels are not available, we train language models as replacements. This achieves semi-supervised training of parameters in our method, which is also discussed in [26]. Therefore, our method can make use of unlabeled data to potentially improve the performance of SLU. Although we mainly focus on using language models as complements to slot filling models, the method can be used for other SLU tasks.

We proposed a step-n-gram model to extract sentence level features. The closest work is a recent method for sequence-to-sequence mapping [32], which uses the last hidden layer activities of RNN for initialization the hidden layer activities of the output sequence. The step-n-gram model is not restricted to the last hidden layer activities, and the exact location is optimally selected via max pooling and maximizing objective functions. The order of step-n-gram model can be increased so that certain sentence-level features are extracted from sequential features with certain time resolutions.

Recently, Xu and Sarikaya [25] have studied joint training of domain identification, intent classification and slot filling. Their approach uses the triangular CRF [33] that has additional variables indicating the domain/intent assignment of the query. Joint training is achieved via modeling the dependency between the the newly introduced variables and the hidden slot labelling sequence. Another novelty is using Convolutional Neural Networks (CNNs) [21] to extract features. Our work differs from theirs in using RNN, incorporating contextual information, and using step-n-gram models for extracting sentence-level features.

3. RECURRENT NEURAL NETWORK BASED JOINT TRAINING

The proposed approach is illustrated in Fig. 1. The bottom part of the figure in a dash round square illustrates a slot filling model using RNN. The output layer $\mathbf{y}_i \in R^K$ produces a probability distribution over possible semantic labels at position i. K is the number of labels. The hidden layer $\mathbf{s}_i \in R^D$ maintains a representation of the sentence history. D is the hidden layer dimension. The input vector $\mathbf{w}_i \in R^N$ has a dimension equal to the vocabulary size N. The values in the hidden and output layers are computed as follows:

$$\begin{aligned} \mathbf{s}_i &= \sigma\left(\mathbf{U}\mathbf{w}_i + \mathbf{W}\mathbf{s}_{i-1}\right) && (1)\\ \mathbf{z}_i &= \mathbf{V}\mathbf{s}_i, && (2)\\ \mathbf{y}_i &= g\left(\mathbf{z}_i\right), && (3) \end{aligned}$$

where

$$\sigma(z) = \frac{1}{1+e^{-z}}, \quad g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}. \qquad (4)$$

and $\mathbf{U}$, $\mathbf{W}$ and $\mathbf{V}$ are the connection weights.

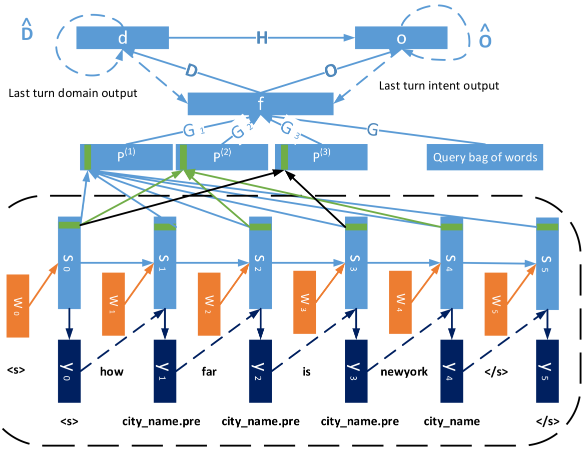

Fig. 1. RNN based contextual SLU.

An original contribution of this work is the introduction of convolution layers that summarize the hidden layer activities from RNN. The convolution operations in [5, 26] are on top of feed-forward NN layer activities that capture the local observations. In contrast, the RNN here captures sequential and sentence level information in the hidden layer activities. Specifically, the hidden layer activities of the last word incurs a memory of hidden layer activities from the previous word instances. Instead of directly using the last hidden layer activity [32], we use max pooling over all of the RNN hidden layer activities. The exact location that has the maximum hidden layer activities is therefore decided via optimizing an objective during training and is selected based on the learned parameters during test. Our method is therefore less ad-hoc as [32]. We plan to conduct further works, possibly on other tasks, to understand its benefits.

We further introduce a concept of step-n-gram max pooling. The "$\mathbf{p}^{(1)}$" is a step-1-gram max pooling that takes all of the hidden layer activities. The "$\mathbf{p}^{(i)}$" is the step-i-gram max pooling that skips $i-1$ hidden layer activities. Using step-n-gram max pooling, each convolution operation generates a fixed dimension feature vector to summarize the hidden layer activities in a certain time-resolution. The step-i-gram max pooling has the same or lower time resolution than the step-j-gram max pooling for $j \leq i$. The step-1-gram max pooling has the highest resolution. Usually, we use a committee of step-1-gram to step-3-gram max pooling.

Together with a feature vector that encodes a bag-of-word frequency, these step-n-gram max pooling activities are fed into NN layer. The bag-of-word frequency feature vector has a dimension of vocabulary size. Each element represents the frequency of occurring a particular word in the current query. The hidden layer activation function in the NN is the sigmoid activation function; i.e.,

$$\mathbf{f} = \sigma\left(\sum_{i=1}^{L} \mathbf{G}_i \mathbf{p}^{(i)} + \mathbf{G}\mathbf{x}\right) \qquad (5)$$

where L is the maximum order for step-n-gram and $\mathbf{x}$ is the bag-of-word feature vector. $\mathbf{G}_i$ and $\mathbf{G}$ are weight matrices.

The domain identification and intent classification use fully connected neural networks and use features from the step-n-gram models. The contextual information is introduced with the use of the past predictions of domain/intent in the last turn. These past predictions are represented as dash arrow curves in the figure. During training, we use the ground-truth of domain/intent assignment of the last turn, but use the predictions during test. This training scheme is also called teacher forcing [34] that usually leads to fast training and may serve as a corrective mechanism during training. The multinomial distribution of these classification is obtained using softmax activation function; i.e.,

$$\mathbf{d} = \mathbf{g}\left(\mathbf{D}\mathbf{f} + \hat{\mathbf{D}}\hat{\mathbf{d}}\right), \tag{6}$$

$$\mathbf{o} = \mathbf{g}\left(\mathbf{O}\mathbf{f} + \mathbf{H}\mathbf{d} + \hat{\mathbf{O}}\hat{\mathbf{o}}\right) \tag{7}$$

where $\mathbf{d} \in R^P$ and $\mathbf{o} \in R^Q$ each denotes the domain and intent classification vector. $\hat{\mathbf{d}} \in R^P$ and $\hat{\mathbf{o}} \in R^Q$ each denotes the previous query domain and intent classification vector. P and Q each denotes the number of domains and intents. $\mathbf{D}$ and $\mathbf{O}$ are the corresponding weight matrices from NN layer to domain and intent output. $\hat{\mathbf{D}}$ and $\hat{\mathbf{O}}$ represent weight matrices from previous domain and intent output to current domain and intent output. $\mathbf{H}$ is the weight matrix from domain output to intent output.

In the proposed RNN based SLU illustrated in Fig 1, there are two hidden layers. The top one is shared by domain and intent classification. The other one is the hidden layer of the bottom RNN that is shared by domain/intent classification and slot labeling. The model size and modeling capability are mainly determined by the size of the RNN hidden layer. Its optimal dimension is dependent on the training data size. Usually, larger vocabulary data requires bigger RNN dimension to have better performance. The dimension of the top level NN layer is mainly determined by the domain and intent label size.

The training uses maximum likelihood criterion and iteratively trains NN and RNN parameters for the joint domain/intent classification and slot labeling. It firstly applies forward computation to predict domains and obtains NN and RNN error signals via backpropagation. With the updated parameters, the training procedure then predicts intents and obtains NN and RNN parameters from intent classification errors. Finally, it updates RNN parameters using error signals from predicting slot labels. Notice that the RNN is updated from error signals in domain/intent classification and slot labeling. The NN is updated from errors signals in domain/intent classification, but is dependent on RNN activities. Therefore, NN and RNN are jointly trained for the three tasks. We use AdaGrad [35] to control the learning rate during training. During test, we apply the proposed SLU in the same way as the standard pipeline. It classifies query domain first, then does intent classification, followed by slot filling.

4. EXPERIMENT

4.1. Data

In this paper, two sets of experiments are carried out to evaluate the proposed SLU method. The first uses ATIS dataset [29, 31]. This dataset is about air travel domain with 22 different intents. There are 893 utterances for testing (ATIS-III, Nov93 and Dec94), and 4978 utterances for training (rest of ATIS-III and ATIS-II). Each word is labeled with one semantic slot. There are 127 slots and 899 unique words. Two types of features can be used on ATIS. The first is the lexicon and the second is the named entity (NE) feature.

ATIS is a single turn data set that doesn't include history turn information. In order to evaluate performances of the proposed contextual SLU in a context-sensitive setup, we conducted the second set of experiments on an internal data set which was derived from Microsoft Cortana live log data. There are 9 domains, which are alarm, calendar, communication, note, ondevice, reminder, weather, places and web. The data set has 112 intents and 71 slots including null (nothing) slot. Training consists of 67,818 queries and 23,823 sessions. Testing set has 6,905 queries and 2,644 sessions.

4.2. Training procedure

Instead of using a validation data set to monitor training progress, we use fixed numbers of iterations. On ATIS data set, the maximum iteration number is set to 50. On the Microsoft Cortana live data, the maximum iteration number is set to 10. Learning rate is adjusted using AdaGrad [35]. The initial learning rate is set to 0.05 on ATIS data and 0.1 for Microsoft Cortana live data. In both sets of experiments, the RNN hidden layer size and NN hidden layer size are set to 100 and 50, respectively. A committee of step-1-gram to step-3-gram max pooling is used to extract query level embedding vectors from RNN hidden layer activities.

4.3. Single Turn Spoken Language Understanding

methods	feature	error rate
SVM	lex+3gram	5.7
SVM	(lex+NE)+3gram	5.6
SVM	lex+2gram	4.4
SVM	(lex+NE)+2gram	4.4
intent+lm	lex	7.4
intent+lm	lex+NE	5.7
intent+slot	lex	7.2
intent+slot	lex+NE	4.8
intent+slot+lm	lex	7.2
intent+slot+lm	lex+NE	4.6

Table 1. Error rates for intent classification by different methods on ATIS. The baseline is SVM. 'intent' stands for training intent classification. 'lm' stands for training language model. 'slot' stands for training slot filling. 'lex' stands for lexicon feature. 'NE' stands for name entity features. '3gram' means the model use unigram, bigram and trigram features.

We conducted experiments on the ATIS dataset to show the effectiveness of joint training. Table 1 reports the error rate for intent classification. The baseline is the linear SVM, which achieves the lowest error rate of 4.4% using lexical, named entity ('NE'), and bigram features ('2gram'). Using trigram ('3gram') feature doesn't improve performances, probably due to overfit. The proposed method of joint training of intent classification and slot labeling using lexicon and named-entity feature achieves 4.8% error rate, which is worse than the linear SVM. However, error rate can be reduced to 4.6% by additionally joint training of language models.

Table 2 reports slot filling performances measured in F1 score using the proposed RNN-based joint training method, in comparison to the published results using CRF, RNN, Long Short-Term-Memory (LSTM) [28] and Recurrent Conditional Random Field (RCRF) [29]. The highest F1 score with lexicon feature only is achieved using LSTM neural networks with 3 order moving average (LSTM-ma3) [28]. The proposed joint training of intent classification and slot filling method achieved 94.63% F1 score, which is close to 94.92% by LSTM-ma3, and preforms better than the conventional RNN based slot filling models [12]. Using the additional name entity ('NE') features, the proposed joint training method using RNN achieved a new state-of-the-art F1 score of 96.83%.

model	feature	F1(%)
CRF [36]	lex	91.09
RNN [12]	lex	94.11
CRF [37]	lex	94.40
RNN joint (this work)	lex	94.63
LSTM [28]	lex	94.85
LSTM-ma3 [28]	lex	94.92
CRF [38]	lex+NE	95.00
RNN [12]	lex+NE	96.60
RCRF [29]	lex+NE	96.46
RNN joint (this work)	lex+NE	**96.83**

Table 2. F1 score (in %) for slot filling on ATIS achieved by different methods using lexicon ('lex') and name entity ('NE') features.

4.4. Contextual Spoken Language Understanding

Microsoft Cortana Live log data has context information in each query. Table 3 reports the domain identification error rates. For comparison, it also includes linear SVM results. To use the domain labels from the previous turn, we use a method proposed in [5] that applies a product feature to model the joint effect of the previous domain label and current turn n-gram features. This method is shown to outperform that using the previous labels directly [5].

Without contextual information, RNN based joint training ('single RNN joint') obtained 10.5% classification error rate, a 13% relative error rate reduction compared to 12.1% by the linear SVM ('single SVM +3gram'). If using true domain assignment of the previous turn ('multi RNN joint'), the proposed method obtained the lowest error rate of 5.4%, a 23% relative error rate reduction compared against SVM using true domain assignment of the previous turn ('multi SVM + 3gram'). However, this gain is reduced to 3% (not statistical significant) when using predicted domain of the last turn.

model	true	predicted
single SVM +3gram	12.1	12.1
single RNN joint	10.5	10.5
multi SVM +3gram	7.0	10.3
multi RNN joint	5.4	10.0

Table 3. Error rate for domain identification on Cortana live log data. SVM and RNN use joint training under single turn and contextual multi-turn situations. 'true' column lists the results using true domain identification of the previous turn. 'predicted' list the results using predicted domain identification.

Table 4 reports intent classification error rate of the proposed methods. In practice, a domain dependent intent classifier is used to detect the intent of the query after a domain label is determined. To simplify the comparison, all the models in Table 4 are based on true domain information. We observed that using contextual information, in particular, the intent classifications from the past turn, reduced error rate of the current turn. For example, classification error rate by RNN ('multi RNN joint') is reduced to 5.0% from 7.2% with true intent classification of the past turn. In general, using predicted intents from the past turn has smaller gain than using true intent of the past turn. Nevertheless, the relative error rate reduction is 19%, from 7.2% by 'single RNN joint' down to 5.8% by 'multi RNN joint'. Compared to SVM with the predicted intent from the past turn ('multi SVM + 3gram'), the relative error rate reduction is 2% that is not statistical significant.

model	true	predicted
singleSVM +3gram	6.9	6.9
multiSVM +3gram	4.8	5.9
single RNN joint	7.2	7.2
multi RNN joint	5.0	5.8

Table 4. Error rate for intent classification on Cortana live log data. SVM and RNN use joint training under single turn and contextual multi-turn situations. 'true' column lists the results using true intent assignment of the past turn. 'predicted' lists the results using predicted intent classification.

Table 5 presents F1 scores of the slot filling results using the proposed RNN-based joint training method, together with results from CRF, RNN and RCRF [29] under the condition that the true domain information is available for each query. On average of 8 domains, RCRF achieves 93.2% F1 score, better than those by RNN and CRF. Given domains, the proposed method does joint training of intent classification with slot filling. Shown in Table 5, the joint training method ('m-RNN') improves F1 score to 94.0% with contextual information ('m-RNN'). Without contextual information, the joint training method ('s-RNN') achieves 93.9% F1 score that outperforms CRF, RNN, and RCRF.

domain	size	CRF	RNN	RCRF	s-RNN	m-RNN
alarm	3815	93.2	93.8	93.9	95.9	94.9
calend	4137	93.7	95.2	95.9	95.5	96.4
commun	9550	91.6	92.5	93.2	93.1	92.9
note	828	76.5	77.0	76.9	80.3	74.9
ondevi	4383	98.2	98.4	98.8	98.4	98.6
places	6166	87.5	88.4	88.3	89.6	90.6
remind	5258	93.3	90.5	91.9	92.2	92.4
weathe	8269	95.7	95.0	94.6	96.3	96.7
avarage		92.6	92.8	93.2	93.9	94.0

Table 5. F1 score (in %) for slot filling on Cortana live log data. For comparison, CRF, RNN and RCRF results are listed. 'domain' column lists all of the tested domains. Column 'size' reports the number of utterances in each domain. 's-RNN' stands for single turn RNN using joint training. 'm-RNN' stands for multi-turn RNN using joint training and contextual information. The bottom row of 'average' reports the average F1 scores weighted by the size of domain.

5. CONCLUSION

We have presented a contextual SLU method that jointly optimizes domain/intent classification and slot filling. This method exploits the context information from the predictions of domain and intent of the past query in a session. Our method uses Recurrent Neural Networks to extract sequential features. On top of the feature, we use a novel step-n-gram model that extracts sentence-level features. We use Convolution Neural Networks for domain and intent classification. We have conducted experiments on the widely used ATIS dataset and obtained new state-of-the-art result for slot filling. In a multi-domain multi-turn contextual spoken language understanding task, the proposed method improves performances over alternative methods. We plan to conduct future research to use more flexible network structures for contextual SLU.

6. REFERENCES

[1] G. Tur, Y. Wang, and D. Hakkani-Tr, *Understanding Spoken Language*, Chapman and HallCRC Press, 2013.

[2] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin, "Liblinear: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[3] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the International Conference on Machine Learning*, 2001, ICML, pp. 282–289.

[4] A. Bhargava, A. Celikyilmaz, D. H. Tur, and R. Sarikaya, "Easy contextual intent prediction and slot detection," in *ICASSP*, May 2013.

[5] P. Xu and R. Sarikaya, "Contextual domain classification in spoken language understanding systems using recurrent nerual network," in *ICASSP*, 2014, p. to appear.

[6] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, "Recurrent neural network based language model," in *INTERSPEECH*, 2010, pp. 1045–1048.

[7] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network based language model," in *ICASSP*, 2011, pp. 5528–5531.

[8] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernocky, "Empirical evaluation and combination of advanced language modeling techniques," in *INTERSPEECH*, 2011, pp. 605–608.

[9] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Cernocky, "Strategies for training large scale neural network language models," in *ASRU*, 2011.

[10] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *Proc. SLT*, 2012, pp. 234–239.

[11] E. Arisoy, T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep 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*, 2012, pp. 20–28.

[12] K. Yao, G. Zweig, M. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *INTERSPEECH*, 2013.

[13] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for language understanding," in *INTERSPEECH*, 2013.

[14] M. Auli, M. Galley, C. Quirk, and G. Zweig, "Joint language and translation modeling with recurrent neural networks," in *EMNLP*, 2013.

[15] H. Schwenk and J.L. Gauvain, "Connectionist language modeling for large vocabulary continuous speech recognition," in *ICASSP*. IEEE, 2002, vol. 1, pp. I–765.

[16] Y. Bengio, R. Ducharme, Vincent, P., and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Reseach*, vol. 3, no. 6, 2003.

[17] H. Schwenk, "Continuous space language models," *Computer Speech and Language*, vol. 21, no. 3, pp. 492–518, 2007.

[18] H.-S. Le, I. Oparin, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Structured output layer neural network language model," in *ICASSP*, 2011, pp. 5524–5527.

[19] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," in *Proceedings of the international workshop on artificial intelligence and statistics*, 2005, pp. 246–252.

[20] T. Mikolov, W.T. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *NAACL-HLT*, 2013.

[21] K. Kavukcuoglu, P. Sermanet, Y-L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun, "Learning convolutional feature hierachies for visual recognition," in *NIPS*, 2010.

[22] R. Sarikaya, G. E. Hinton, and B. Ramabhadran, "Deep belief nets for natural language call-routing," in *ICASSP*, 2011, pp. 5680–5683.

[23] L. Deng, G. Tür, X. He, and D. Z. Hakkani-Tür, "Use of kernel deep convex networks and end-to-end learning for spoken language understanding.," in *SLT*, 2012, pp. 210–215.

[24] G. Tür, L. Deng, D. Hakkani-Tür, and X. He, "Towards deeper understanding: Deep convex networks for semantic utterance classification," in *ICASSP*, 2012, pp. 5045–5048.

[25] P. Xu and R. Sarikaya, "Convolutional neural network based triangular CRF for joint intent detection and slot filling," in *ASRU*, 2013, pp. 78–83.

[26] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the International Conference on Machine Learning*, 2008, pp. 160–167.

[27] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[28] K. Yao, B. Peng, Y. Zhang, D. Yu, G. Zweig, and Y. Shi, "Spoken language understanding using long short-term memory neural networks," in *Proceedings of IEEE workshop on Spoken Language Technology*, 2014.

[29] K. Yao, B. Peng, G. Zweig, D. Yu, X.Li, and F. Gao, "Recurrent conditional random field for language understanding," in *ICASSP*, 2014.

[30] G.Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," in *INTERSPEECH*, 2013.

[31] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The atis spoken language systems pilot corpus," in *Proceedings of the Workshop on Speech and Natural Language*, 1990, pp. 96–101.

[32] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *arxiv*, 2014.

[33] M. Jeong and G. G. Lee, "Triangular-chain conditional random fields," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 16, pp. 1287–1302, 2008.

[34] R. J. William and D. Zipser, "Gradient-based learning algorithms for recurrent networks and their computational complexity," in *Backpropagation: Theory, Architectures, and Applications*, 1995, pp. 433–486.

[35] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[36] L. Deng, G. Tur, X. He, and D. Hakkani-Tur, "Use of kernel deep convex networks and end-to-end learning for spoken language understanding," December 2012, SLT.

[37] G. Tur, D. Hakkani-Tür, L. Heck, and S. Parthasarathy, "Sentence simplification for spoken language understanding," in *ICASSP*, 2011.

[38] C. Raymond and G. Riccardi, "Generative and discriminative algorithms for spoken language understanding," in *INTERSPEECH*, 2007.