INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

A New Pre-training Method for Training Deep Learning Models with Application to Spoken Language Understanding

Asli Celikyilmaz, Ruhi Sarikaya, Dilek Hakkani-Tur, Xiaohu Liu, Nikhil Ramesh, Gokhan Tur

Microsoft

asli|dilek|gokhan.tur@ieee.org, ruhi.sarikaya|derek.liu|nikhilr@microsoft.com

Abstract

We propose a simple and efficient approach for pre-training deep learning models with application to slot filling tasks in spoken language understanding. The proposed approach leverages unlabeled data to train the models and is generic enough to work with any deep learning model. In this study, we consider the CNN2CRF architecture that contains Convolutional Neural Network (CNN) with Conditional Random Fields (CRF) as top layer, since it has shown great potential for learning useful representations for supervised sequence learning tasks. The proposed pre-training approach with this architecture learns the feature representations from both labeled and unlabeled data at the CNN layer, covering features that would not be observed in limited labeled data. At the CRF layer, the unlabeled data uses predicted classes of words as latent sequence labels together with labeled sequences. Latent labeled sequences, in principle, has the regularization effect on the labeled sequences, yielding a better generalized model. This allows the network to learn representations that are useful for not only slot tagging using labeled data but also learning dependencies both within and between latent clusters of unseen words. The proposed pre-training method with the CRF2CNN architecture achieves significant gains with respect to the strongest semi-supervised baseline.

Index Terms: Unsupervised pre-training, semi-supervised slot filling, convolutional neural network, triangular CRF.

1. Introduction

There has been tremendous investment on personal digital assistants (PDAs) and agents in recent years [1]. Spoken language understanding (SLU) has emerged as the main interface to interact with the PDAs. Robust SLU systems consider three main tasks to extract meaning from user utterances: domain and intent classification and slot filling [2]. Building highly accurate models for each of these tasks is of critical importance for improved user experience and fast task completion with PDAs.

Recently, deep learning based techniques have been heavily applied to speech and language understanding problems including language modeling [3], text processing [4], multi-modal learning [5, 6], to name a few. The strength of neural networks (NN) is in learning the feature representations from large amounts of unlabeled training data and using them for task specific models. Such models, in theory, should yield the same performance with less amount of labeled training data. Recent research on using the labeled and unlabeled data has reported successful results using two step learning methods, where initial representations are obtained from unlabeled data through pre-training (e.g., auto-encoders, hidden unit CRFs) [7, 8], which are later used as initial representations in task specific supervised models. In more recent work [9, 10], a simultaneous learning of the network structure using supervised and unsupervised training have been investigated. For instance, in [9] a NN classifier model is presented where the network weights at each layer are trained by minimizing the combined loss function of an auto-encoder and a classifier. In [10], a joint pre-training and NN classification is presented that learns the network from both labeled and unlabeled data. They simultaneously predict pseudo-labels for the unlabeled data, and update the weights based on the loss function measured by the difference between the actual and pseudo labels against the predicted labels.

In this work, we present a new pre-training method for deep NNs in general, CNNs in particular, for slot tagging by way of semi-supervision. Our goal is to jointly learn the network structure from large unlabeled data, while learning to predict the task specific semantic (slot) tags from labeled sequences. Extending the supervised CNN with CRF architecture of [11], we use CNN as the bottom layer to learn the feature representations from labeled and unlabeled sequences. At the top layer, we use two CRF structures. The first CRF model weights are updated only with the labeled training data where the output sequences are comprised of slot tag sequences. The second CRF model weights are updated with the unlabeled utterances, where the latent class labels are used as output sequences. This allows the network to simultaneously learn the transition and emission weights for slot tagging and class labeling of the words in utterances in a single model. The key challenge is to find "representative" output label sequences (i.e., latent class sequences) for the unlabeled utterances, which in turn should be related to the task of interest. Thus, we first cluster the words of the unlabeled data and use the predicted cluster IDs of each word as latent sequence tags. This enables us to build a structure, which we denote as CNN2CRF. This structure consists of two supervised CRF models with different output sequences, i.e, slot tag sequences for labeled data, and latent cluster id sequences for unlabeled data. With our proposed method, we are able to show that the proposed method with the CNN2CRF network architecture achieves gains over strong baselines.

In the next, we present the the CNN2CRF architecture provinding details on the CNN and CRF layers as well as the latent clustering. The experiments and results are presented in section 3 followed by the conclusion.

2. Pre-Training for CNNs

The slot filling task of SLU involves extracting relevant semantic constituents from natural language utterances. It is often formulated as a sequence labeling task, where a sequence of labels are jointly assigned to the words of a sentence. Recent NN models use a variety of structures for slot filling: a recurrent neural network (RNN) [12, 13, 14], or a long short term memory

http://dx.doi.org/10.21437/Interspeech.2016-512

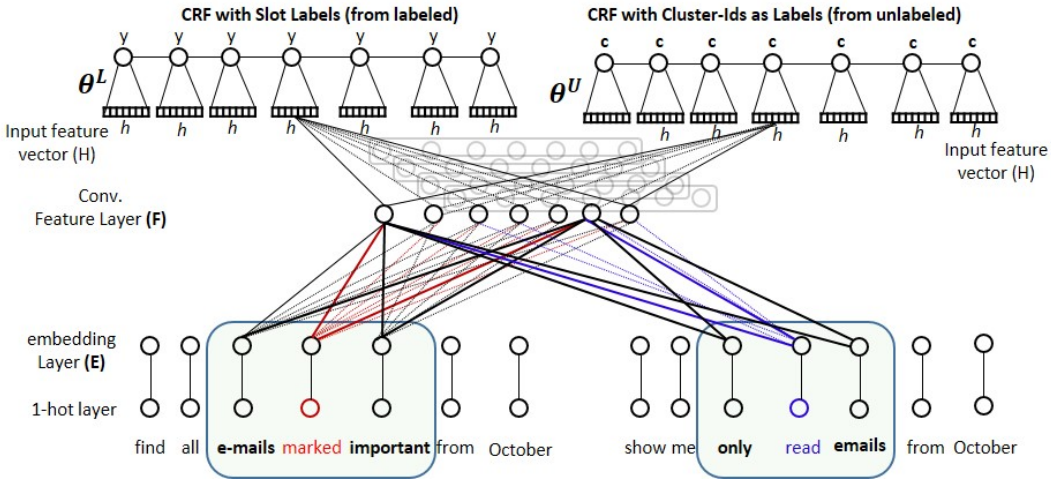

Figure 1: *CNN with two CRF layers (CNN2CRF) that use labeled and unlabeled data respectively.The network has one convolutional layer. The width of the feature transformation layers are 3 with F number of filters (F=7 is chosen for demonstration). Pooling of the features (e.g., max-pooling) generate feature layer h which is fed to the CRF layers. Left CRF learns the weights θ^L on labeled data to generate slot tag sequences, whereas the CRF on the right learns the weights θ^U on unlabeled data sequences to generate latent class label (cluster-id) sequences. Each CRF shares the same features from CNN layer.*

(LSTM) [15] or a CRF model stacked over a CNN bottom layer [11], which show improved performances over baselines such as CRF. CNNs have enjoyed wide success in the last few years in several domains including images, video, audio and natural language processing. In this paper, although we use CNN as the feature representation learning layer- due to its speed and efficiency for learning representations, it can be easily replaced by a recurrent neural network (e.g., LSTM[16] or GRU[17]), making our approach generic enough to work with other state of the art deep learning methods. This will be investigated as part of the future work. Figure-1 shows an overview of the proposed CNN2CRF architecture, which comprises of three main components: a CNN as encoder and two top layers of CRFs as decoder. In the following, we explain our two layer-3 component model in detail.

2.1. CNNs for Learning Features for Slot Tagging

In Figure-1 (bottom), 1-dimensional CNN layer uses labeled and unlabeled word sequences to extract continuous-valued features. It can handle input sequences (as sequence of words) of varying lengths. They utilize layers with convolving filters that are applied to local features [18], which are combination of words or representation of words obtained from pre-training (e.g., vector representations of words, also known as word embeddings). Every filter performs convolution on the sentence matrix and generates feature maps. First, the words are embedded into low-dimensional vectors. Next, convolutions over the embedded word vectors are performed using multiple filter sizes, $f_{j=1,..,F}$. For example, sliding over c=3 words at a time, using the first filter f_j the feature vector for center word i $'marked'$ is: $h_{ij} = \sum_{jd}(e_{id}['emails', \textbf{'marked'}, 'important'] * f_{jd})$, where d is the $d = c{*}embeddingdimension$. Figure-1 is only showing the calculation of the center words 'marked' from labeled and 'read' from unlabeled data. Next, max-pool or sum-pool is applied on the result of the convolutional layer into a long feature vector. Here, we use sum-pooling, which generated better results in our experiments. Random dropout on the feature vector dimensions is used for regularization. The automatically learned feature vectors of each word through CNN is shared by the top CRF layers to predict the best sequences.

2.2. The CNN2CRF with Pre-Training

The two top CRF models are essentially the same as 1^{st} order CRF models- the only difference is that the features from the word sequences are automatically extracted from the bottom CNN layers, instead of user defined features and n-grams. The conditional distribution of the first CRF with slot tag labels (Figure-1 top-left) is given by:

$$P(Y|S^L) = \frac{e^{\sum_i (t(Y_{i-1},Y_i) + \sum_j h_{ij}(S_i,E,F)\theta^L_j(Y_i))}}{\sum_{Y'} e^{\sum_i (t(Y'_{i-1},Y'_i) + \sum_j h_{ij}(S_i,E,F)\theta^L_j(Y'_i))}} \quad (1)$$

where S^L is the words sequence from labeled data, $t(Y_{i-1}, Y_t)$ is the tag transition score from Y_{i-1} to Y_i. $h_{ij}(S_i, E, F)$ denotes the j^{th} element in the feature vector extracted out of the c-gram window centered at S_i using the word-embedding E and feature transformations filters F of the center word and its c window. $\theta^L_j(Y_i)$ is the corresponding feature weight associated with the tag Y_i. The second CRF is similar to the first, but it only uses latent class labels:

$$P(C|S^U) = \frac{e^{\sum_i (t(C_{i-1},C_i) + \sum_j h_{ij}(S_i,E,F)\theta^U_j(C_i))}}{\sum_{C'} e^{\sum_i (t(C'_{i-1},C'_i) + \sum_j h_{ij}(S_i,E,F)\theta^U_j(C'_i))}} \quad (2)$$

where $\theta^U_j(C_i)$ is the corresponding feature weight associated with the class C_i. One of the CRFs is performing sequence tagging on labeled training data using slot tags as labels, the other one is also performing sequence tagging on unlabeled data using class IDs as tags. Hence, the multi-task CNN2CRF architecture has two separate loss terms both of which tries to minimize the negative conditional log-likelihood of the true parse according to the model, $\{\theta^L, \theta^U\}$. The labeled loss term (LL) uses the true slot tags $Y^{(l)}$ given a sentence word sequence

$S^{(l)}= w_1, ..., W_T$ from the set of N_L sequences.

$$LL = -\sum_{l=1}^{N_L} logP_{\theta^L}(Y^{(l)}|S^{(l)};\theta^L) \quad (3)$$

wheres the unlabeled loss term (UL) uses the predicted (latent) class labels $C^{(u)}$ given unlabeled word sequence $S^{(u)}$=$w_1, \cdots, W_T$ from the set of N_U sequences.

$$UL = -\sum_{u=1}^{N_U} logP_{\theta^U}(C^{(u)}|S^{(u)};\theta^U) \quad (4)$$

Because the total number of labeled data and unlabeled data is quite different and the training balance between them affect the prediction of slot tags for a given utterance, we weight in the unlabeled loss using the following criterion:

$$loss = LL + \alpha(t) * UL \quad (5)$$

where $\alpha(t)$ is a balancing coefficient. The proper selection of $\alpha(t)$ is important in the network's performance. Rather than iterating over a range of values and setting the optimum value based on the performance of the network on a development set (which is a common approach), we follow the deterministic annealing process of [19] in which the network starts with small $\alpha(t) \sim 0.0001$ and is slowly increased following the criterion:

$$\alpha(t) = \begin{cases} 0 & t < T_1 \\ \frac{t-T_1}{T_2-T_1} & T_1 \geq t < T_2 \\ \alpha_f & T_2 \geq t \end{cases} \quad (6)$$

with $\alpha_f = 3$, T_1=100, T_2=600. This is expected to avoid the poor local minima, should the unlabeled data labels are too noisy or the size of the unlabeled data is too large.

The training of the topmost layer is same as training a standard CRF model. The bottom layers are trained using the well-known back propagation algorithm which is essentially the chain rule of derivatives. We take the derivative of the loss function with respect to each $h_{ij}(S^{(u|l)}, E, F)$ at the top layer ($(u|l)$,:labeled or unlabeled) with respect to E and F by applying the chain rule to the CNN layer.

Latent Class Labels: The class labels of the unlabeled utterances are initially captured in the following way: We convert each word in the unlabeled data into vector representations such that the relationship between two vectors mirrors the linguistic relationship between the two words. Hence, we learn the mapping $V \rightarrow R^D : w \rightarrow \overrightarrow{w}$ using a popular unsupervised embedding method, namely GloVe [20], which captures the relevant information about the relation of words from the original co-occurrence matrix[1]. We later cluster the word embeddings into K classes based on the nearest neighbor approach and assign each word a cluster id, w=$k \in K$. We use the class labels as latent sequence tags for the unlabeled training data.

2.3. CNNCRF with Pseudo Labels

As a benchmark, we experiment our joint architecture with unlabeled data without class labels, similar to [10]. At network contruction time, we learn the CNN bottom network on both unlabeled and labeled data, similar to our proposed approach. Unlike our proposed CNN2CRF, for the unlabeled data, we do not build a second CRF. Instead, we generate pseudo-slot tags for the unlabeled data as if they were true slot sequences and we refer this network as CNNCRF-P using the following loss:

$$loss' = -\sum_{l=1}^{N_L} logP_\theta(Y^{(l)}|S^{(l)};\theta) - \alpha(t)\sum_{u=1}^{N_U} logP_\theta(\hat{Y}^{(u)}|S^{(u)};\theta) \quad (7)$$

where the first term is same as in LL, the second term contains the $\hat{Y}^{(u)}$ indicating the predicted tag sequences of the unlabeled utterances from the single CRF layer and the θ is learnt from all the training data (dropping L and U superscripts since we have only one CRF top model). We choose $\alpha(t)$=0 until $t < T_1$ so the network learns the weights from labeled data initially. At t=T_1, we start decoding the unlabeled data and use the predicted tags $\hat{Y}^{(j)}$ as pseudo labels decoded from the last weight update.

3. Experiments

Datasets: The internal corpora used for training and testing consists mostly of logs of spoken utterances or typed input collected from real users of Cortana Microsofts PDA. We focus here on 9 domains, as shown in Table 1. The scenarios involve the users interacting with the devices (e.g., phone, tablet, desktop, console, etc.) by voice with a system that can perform a variety of tasks in relation to each domain, including (among others) browsing, searching, playing, purchasing, setting up alarm, setting device configuration parameters, etc. We use the transcribed text utterances obtained from ASR engine. Per domain, we used 100K and 20K utterances for train and test dataset respectively. The unlabeled training data is 10 times as large as the labeled training data.

Models: For benchmark purposes we use (*i*) the linear chain CRF, (*ii*) supervised CNNCRF [11], (*iii*) supervised RNNCRF[2] [23] with n-gram features (up to 3-grams) and (*iv*) supervised CNNCRF with pre-training (CNNCRF-Pre), where we used the same GloVe embeddings as initial word embeddings. We compare the benchmark models, which do not use unlabeled data, against the proposed CNN2CRF that uses class labels for unlabeled data and CNNCRF-P that uses pseudo labels.

We use stochastic gradient descent to learn the model parameters. We use development set to track the training process. We started the learning rate from 0.005 which halves every time the result on the dev set ceases to improve for 5 consecutive iterations. For stopping criteria we used the learning rate which stopped the training process if the rate falls below 0.0001. We initialized all the network parameters randomly. We chose to use rectifier activation function at input feature layer h along with dropout for regularization (with dropout probability 0.5).

We minimally swept the parameters for the hidden layers, convolutional n-gram windows and feature filter sizes. Unless stated otherwise, we used 100 dimensional embedding vectors, 100 for hidden feature layers and 3-gram window centered around the current position. We only used lexical features.

3.1. Experiment-1: Overall Slot Performance

In this experiment, we compare the proposed CNNCRF to the benchmark models on the slot filling performance. Table-1 shows the benchmark results in F-score on 9 different domains. The average scores of the domains per model is shown in the last row. Although on average, CNN2CRF outperforms all the other models in almost all domains as well as the average F-score, the improvement may not be considered statistically significant

[1]CBOW [21] or CCA [22] can be used for embedding learning.

[2]code:https://rnnsharp.codeplex.com/

Table 1: *Comparison of F-Scores of NN models on different domains. The last line indicates the average F-Score on test data averaged over 9 domains. The winning model F-Score is shown in bold.*

Domain	CRF	CNNCRF	RNNCRF	CNNCRF-Pre	CNN2CRF	CNNCRF-P
alarm	96.21	96.17	96.18	96.08	**96.74**	96.09
calendar	91.33	91.41	91.21	91.23	**91.62**	90.23
notes	88.32	88.89	87.82	88.80	**89.00**	88.29
media	93.67	94.30	93.74	93.34	**94.96**	93.84
devices	92.64	**94.51**	94.25	94.33	94.43	94.25
flights	93.30	**94.69**	92.86	94.65	93.76	93.24
hotels	93.47	93.25	92.01	93.11	**94.00**	93.38
sports	82.90	83.68	82.36	83.26	**83.93**	83.11
travel	87.18	89.20	87.60	89.20	**89.44**	88.94
average	91.11	91.90	90.89	91.46	**91.93**	91.55

(based on t-test) for some domains (e.g. sports, travel) Nevertheless, in some domains the proposed CNN2CRF improves the F-Score to more than 0.5 F-score points, e.g., alarm and media, which is statistically significant given the results (t-test, $p<0.1$).

It is worth to note that, we used strong baselines in this experiment attributing to comparable results in Table-1. To further justify the performance improvement of the proposed CNN2CRF model on statistically significant domains against the strongest baseline, CNNCRF, we investigated the prediction errors of both models. We observe that the CNN2CRF is able to tag the out-of-vocabulary words in testing data, which doesn't appear in labeled training data, correctly while CNNCRF missed them half of the times. Also, the CNN2CRF makes less mistakes on tagging the words that appear less frequently (less than 3 times) given a window of words. Some examples are shown in Table 2. For instance, although *'dvd'* is one of the frequent words in devices domain, the training data did not include *"to start dvd"* as frequently as *"to play dvd"* or *"files on dvd"* causing the baseline models fail, which the CNN2CRF models tags correctly.

Table 2: *Samples of slot tag results. Underline words are tagged differently by CNN2CRF and CNNCRF models, shown in the last two columns.* ***Bolded*** *are correctly predicted slot tags.*

utterance	CN2CRF	CNNCRF
who is mount everest	**place-name**	actor-name
how to start dvd	**device-name**	storeage
open blank word document	**doc-type**	O

These results can be easily attributed to the fact that the CNN2CRF learns features for words from a richer context because the CNN layer uses large number of unlabeled data. In addition, the CRF layer learns the context conditioned not only on the slot tags but also on the latent class tags, which have the regularization affect on learning representative word context features.

3.2. Experiment-2: Scalability

In this experiment, given that we can build the CNN2CRF network with labeled and unlabeled data, we wanted to test how much labeled data would be sufficient to obtain the same performance. For this, we start with 10% labeled training data to build the network and use all the unlabeled traning data and measure the networks performance. We then incrementally add more labeled training data - doubling the size of the training data at each experiment and report results on test data. We compare CNN2CRF and CNNCRF-P as well as an additional model which we call Ghost CNNCRF. The Ghost CNNCRF uses unlabeled and labeled training data similar to CNN2CRF and CNNCRF-P, but does not use the unlabeled data for learning the CRF layer as the other two. The unlabeled data is only used to learn the parameters of the CNN layer.

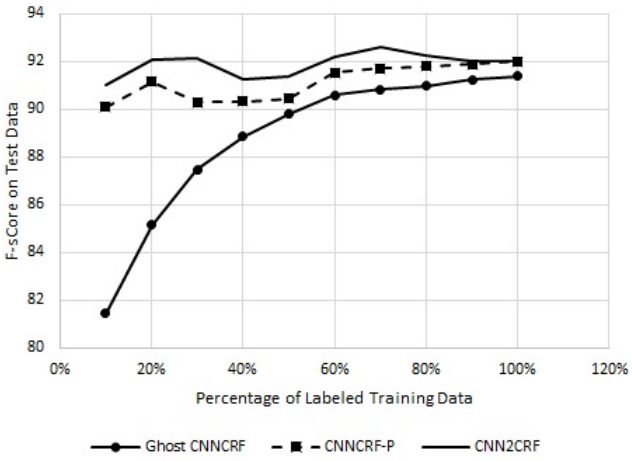

Figure 2: *Scalability Analysis: how much labeled data we should use together with unlabeled data to reach the performance of a model when all labeled data is used for training.*

Figure-2 shows the results in a graph. The striking thing to note is that, when unlabeled data is used to train the CRF structures as in CNNCRF-P and CNN2CRF, with only 10-20% labeled training data one can achieve almost 90% F-score. The CNN2CRF does actually show a better performance than the CNNCRF-P as it uses the pre-trained class labels as labels to learn a secondary CRF model, which demonstrates the impact of the pre-training technique presented in this paper.

4. Conclusions

We have described a new pre-training approach for convolutional neural network with conditional random fields approach that uses the latent class labels as sequence tags for the unlabeled utterances. This is, in effect, a regularization on the word-slot tag relations regulated by the word-class relations. Our network achieves comparable performance on slot filling tasks for real conversational agent data without extensive amount of manually labeled training data.

Our recent experiment (after the paper has submitted) has shown that using NN structures other then CNN's such as RNNs provide additional information in learing the latent features for the multiple tasks. We were able to show improvements on ATIS dataset.We will inveistigate this further on the other datasets presented in this paper. In addition, one interesting direction is to build an end-to-end framework where the learning of the latent class labels from unlabeled data are also handled by the same network. We leave these as future work.

5. Acknowledgements

We would like to thank Puyang Xu, and Ye-Yi Wang for useful discussions about the presented work.

6. References

[1] R. Sarikaya, "The technology powering personal digital assistants http://interspeech2015.org/program/keynotes/," *Proc of INTERSPEECH (Keynote)*, 2015.

[2] G. Tur and R. M. Eds, "Spoken language understanding systems for extracting semantic information from speech," *New York, NY: John Wiley and Sons*, 2011.

[3] E. Arisoy, T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep neural network language models," *Proc. of ACL*, 2012.

[4] X. Zhang and Y. LeCun, "Text understanding from scratch," *Proc. of Computation and Language - arXiv:1502.01710*, 2015.

[5] N. Srivastava and R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," *Proc. of NIPS*, 2012.

[6] R. Kiros, Richard, S. Zemel, and R. Salakhutdinov, "Multimodal neural language models," *Proc. of ICML*, 2014.

[7] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, 2006.

[8] Y.-B. Kim, K. Stratos, and R. Sarikaya, "Pre-training of hidden-unit crfs," *Proc. of ACL*, 2015.

[9] M. Ranzato and M. Szummer, "Semi-supervised learning of compact document representations with deep networks," *Proc. of the 25th International Conference on Machine Learning, ICML*, 2008.

[10] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," *Proc. of the 25th International Conference on Machine Learning, ICML*, 2013.

[11] P. Xu and R. Sarikaya, "Convolutional neural network based triangular crf for joint intent detection and slot filling," *Proc. of ASRU*, 2013.

[12] E. Simonnet, N. Camelin, P. Delglise, and Y. Estve, "Exploring the use of attention-based recurrent neural networks for spoken language understanding," *Proc. of the NIPS Workshop on Machine Learning for SLU and Interaction*, 2015.

[13] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, "Using recurrent neural networks for slot filling in spoken language understanding," *Proc. of IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2015.

[14] Y. Dauphin, G. Tur, D. Hakkani-Tur, and L. Heck, "Zero-shot learning and clustering for semantic utterance classification," *Proc. of International Conference on Learning Representations (ICLR)*, 2014.

[15] K. Yaho, B. Peng, Y. Zhang, and D. Yu, "Spoken language understanding using long short-term memory neural networks," *Proc. of the Spoken Language Technology Workshop (SLT)*, 2014.

[16] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *CoRR, abs/1508.01991*, 2015.

[17] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representatiosn using rnn encoder-decoder for statistical machine translation," *Proc. of EMNLP*, 2014.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. of IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[19] Y. Grandvalet and Y. Bengio, "Entropy regularization," *In Semi Supervised Learning*, pp. 151–168, 2006.

[20] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," *Proc. of EMNLP*, 2014.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Proc. of the NIPS*, 2013.

[22] P. S. Dhillon, J. Rodu, D. P. Foster, and L. H. Ungar, "Two step cca: A new spectral method for estimating vector models of words," *Proc. of ICML*, 2012.

[23] K. Yao, B. Peng, G. Zweig, D. Yu, X. Li, and F. Ga, "Recurrent conditional random field for language understanding," *Proc of ICASSP*, 2014.