# ZERO-SHOT LEARNING OF INTENT EMBEDDINGS FOR EXPANSION BY CONVOLUTIONAL DEEP STRUCTURED SEMANTIC MODELS

*Yun-Nung Chen*[⋆†] *Dilek Hakkani-Tür*[†] *Xiaodong He*[†]

[⋆]Carnegie Mellon University, Pittsburgh, PA, USA
[†]Microsoft Research, Redmond, WA, USA

yvchen@cs.cmu.edu, dilek@ieee.org, xiaohe@microsoft.com

## ABSTRACT

The recent surge of intelligent personal assistants motivates spoken language understanding of dialogue systems. However, the domain constraint along with the inflexible intent schema remains a big issue. This paper focuses on the task of intent expansion, which helps remove the domain limit and make an intent schema flexible. A convolutional deep structured semantic model (CDSSM) is applied to jointly learn the representations for human intents and associated utterances. Then it can flexibly generate new intent embeddings without the need of training samples and model-retraining, which bridges the semantic relation between seen and unseen intents and further performs more robust results. Experiments show that CDSSM is capable of performing zero-shot learning effectively, e.g. generating embeddings of previously unseen intents, and therefore expand to new intents without re-training, and outperforms other semantic embeddings. The discussion and analysis of experiments provide a future direction for reducing human effort about annotating data and removing the domain constraint in spoken dialogue systems.

***Index Terms***— zero-shot learning, spoken language understanding (SLU), spoken dialogue system (SDS), convolutional deep structured semantic model (CDSSM), embeddings, expansion.

## 1. INTRODUCTION

With the surge of smart devices, recent efforts have focused on developing virtual personal assistants (e.g. Apple Siri, Microsoft Cortana, Google Now, Amazon Echo, etc), where spoken language understanding (SLU) is a key component of a spoken dialogue system (SDS), that parses user utterances into corresponding intents and associated semantic slots [1]. Typically all domains are implemented independently, where training intent detectors and slot taggers require manually annotated data [2, 3, 4]. However, the intents are usually predefined and inflexible to expand. For example, an SLU component designed for handling only the air travel reservation task cannot handle new intents such as checking the flight status or making hotel reservations. Traditionally, a standard solution is to redesign a semantic schema adding new intents with associated slots to cover the new intents, which requires human effort for annotation and model re-training [5]. These issues remain the biggest challenge for SDS [6, 7].

To address the issue about intent expansion, this paper investigates zero-shot learning of embeddings for unseen intents, e.g. learning a model to generate semantic embeddings for unseen intents without manually annotated data and without model re-training. The idea is that although the intents "find_movie" and "find_weather" belong to movie and weather domains respectively, they both contain the semantics about "find", so such information should allow us to learn unseen intent representations based on the trained model, which benefits from semantics of other domains. Then the newly learned intent representations can help intent expansion in a flexible fashion.

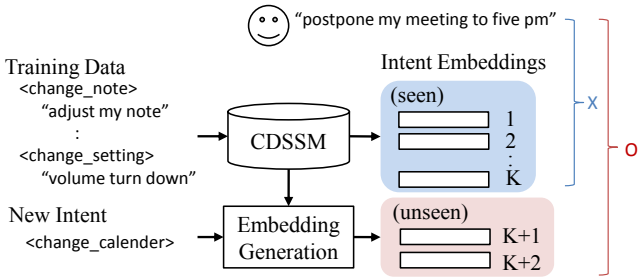


**Fig. 1**. The proposed intent expansion framework. The utterances in training data are used for model training, and the CDSSM generates embeddings for both seen intents (blue block) and unseen intents (pink block) without model re-training in order to predict intents.

Previous work investigated the bootstrapping of SLU models for a new application by re-using the annotated intent data from other applications and creation of an intent library for that purpose [8, 9]. Recently, El-Kahky et al. showed that leveraging knowledge graphs and click logs can determine semantically similar slots to transfer intents across domains for extending the domain coverage [6, 10]. Kim et al. also proposed to automatically generate the mapping between semantic slots across domains by learning semantic label embeddings [7]. Both studies implied that semantics from different domains can be shared and finding the connection helps domain adaptation and expansion. Instead of modeling the relations between intents from different domains, this paper applies convolutional deep structured semantic models (CDSSM) to directly learn complete intent embeddings using intents available in the training data, and then when expanding to new intents, the trained CDSSM is used to construct the representations of new intents based on semantics from the seen data. The assumption is that although the intents are usually treated as categorized identifiers, they usually have meaningful names that contain general semantics. Therefore, the model trained to capture the semantics of the intents can generalize to model new intents unseen before. Finally the new intents can be included in the dialogue systems without the need of new, associated training samples, and model training, reducing human effort and time for intent expansion and making SDS more practical.

In this paper, we treat intent detection as an utterance classification task, where each user utterance corresponds to an intent. Recent studies used CDSSM to map questions into relation-entity triples for question answering [11, 12], which motivates us to use CDSSM for

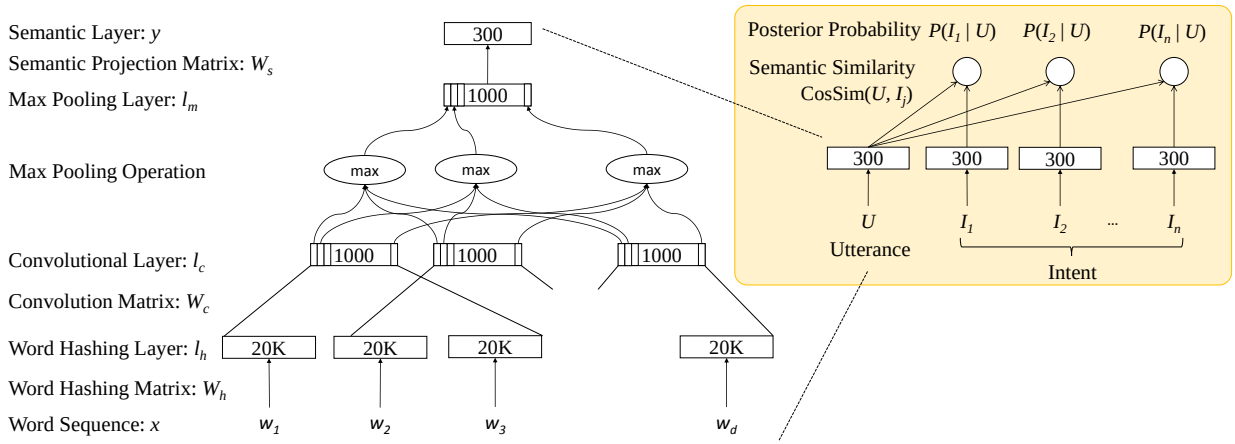


**Fig. 2**. Illustration of the CDSSM architecture for the predictive model.

capturing relations from intent-utterance pairs [13], while vector representations for intents and utterances can be learned by CDSSM. Considering that several studies investigated embedding vectors as features for training task-specific models [14, 15, 16, 1], the representations of intents and utterances can incorporate more informative cues from large data. Hence, this paper focuses on taking CDSSM features to help detect intents (including seen and unseen intents), and the framework is shown in Fig. 1. First, we train a CDSSM for learning intent embeddings, as described in Section 2. Then we generate embeddings for new intents and utilize them to perform intent prediction described in Section 3. Finally Section 4 discusses experiments, and Section 5 concludes the paper.

## 2. CONVOLUTIONAL DEEP STRUCTURED SEMANTIC MODELS (CDSSM)

### 2.1. Architecture

The model is a deep neural network with the convolutional structure, where the architecture is illustrated in Fig. 2 [15, 17, 18, 19, 20]. The model contains: 1) a word hashing layer obtained by converting one-hot word representations into tri-letter vectors, 2) a convolutional layer that extracts contextual features for each word with its neighboring words defined by a window, 3) a max-pooling layer that discovers and combines salient features to form a fixed-length utterance-level feature vector, and 4) a semantic layer that further transforms the max-pooling layer to a low-dimensional semantic vector for input utterances.

**Word Hashing Layer** $l_h$. Each word from a word sequence (i.e. an utterance or an intent name) is converted into a tri-letter vector [18]. For example, the tri-letter vector of the word "#email#" (# is a word boundary symbol) has non-zero elements (value equals one in this case) for "#em", "ema", "mai", "ail", and "il#" via a word hashing matrix $W_h$. Then we build a high-dimensional vector $l_h$ by concatenating all word tri-letter vectors. The advantages of tri-letter vectors include: 1) unseen intents and OOV words can be represented by tri-letter vectors, where the semantics can be captured based on the subwords such as prefix and suffix; 2) the tri-letter space is smaller, where the total number of tri-letters seen in the training data in our experiments is about 18.8K. Therefore, incorporating tri-letter vectors improves the representation power of word vectors and also flexibly represents intents for the purpose of intent expansion while keeping the size small.

**Convolutional Layer** $l_c$. A convolutional layer extracts contextual features $c_i$ for each target word $w_i$, where $c_i$ is the vector concatenating the tri-letter word vector of $w_i$ and its surrounding words within a window (the window size is set to 3 in our experiment). For each word, a local feature vector $l_c$ is generated using a `tanh` activation function and a shared linear projection matrix $W_c$:

$$l_{ci} = \tanh(W_c^T c_i), \text{ where } i = 1, ..., d, \quad (1)$$

where $d$ is the total number of windows.

**Max-Pooling Layer** $l_m$. The max-pooling layer forces the network to only retain the most useful local features by applying the max operation over each dimension of $l_{ci}$ across $i$ in (1),

$$l_{mj} = \max_{i=1,...,d} l_{ci}(j). \quad (2)$$

The convolutional and max-pooling layers are able to capture prominent words of the word sequences [15, 17]. As illustrated in Fig. 2, if we view the local feature vector $l_{c,i}$ as a topic distribution of the local context window, e.g., each element in the vector corresponds to a hidden topic and the value corresponds to the activation of that topic, then taking the max operation at each element keeps the max activation of that hidden topic across the whole sentence.

**Semantic Layer** $y$. The global feature vector $l_m$ in (2) is fed to feed-forward neural network layers to output the final non-linear semantic features $y$ as the output layer.

$$y = \tanh(W_s^T l_m), \quad (3)$$

where $W_s$ is a learned linear projection matrix. The output semantic vector can be either an utterance embedding $y_U$ or an intent embedding $y_I$.

### 2.2. Training Procedure

The seen data containing utterances and associated intents is used for training the model. The idea of this model is to learn the embeddings for both utterances and intents such that utterances with the same intents can be close to each other in the continuous space, as shown in Fig. 2. Below we define a semantic score between an utterance $U$ and an intent $I$ using the cosine similarity between their embeddings, $y_U$ and $y_I$:

$$CosSim(U, I) = \frac{y_U \cdot y_I}{\|y_U\| \|y_I\|}. \quad (4)$$

#### 2.2.1. Predictive Model

The posterior probability of a possible intent given an utterance is computed based on the semantic score through a softmax function,

$$P(I \mid U) = \frac{\exp(CosSim(U, I))}{\sum_{I'} \exp(CosSim(U, I'))}, \quad (5)$$

where $I'$ is an intent candidate.

For model training, we maximize the likelihood of the correctly associated intents given all training utterances. The parameters of the model $\theta_1 = \{W_c, W_s\}$ are optimized by an objective:

$$\Lambda(\theta_1) = \log \prod_{(U, I^+)} P(I^+ \mid U). \quad (6)$$

The model is optimized using mini-batch stochastic gradient descent (SGD) [18].

#### 2.2.2. Generative Model

Similarly, we can estimate the posterior probability of an utterance given an intent using the reverse setting,

$$P(U \mid I) = \frac{\exp(CosSim(U, I))}{\sum_{U'} \exp(CosSim(U', I))}, \quad (7)$$

which is the generative model that emits the utterances for each intent. Also, the parameters of the model $\theta_2$ are optimized by an objective:

$$\Lambda(\theta_2) = \log \prod_{(U^+, I)} P(U^+ \mid I). \quad (8)$$

The model can be obtained similarly and performs a reversed estimation for the relation between utterances and intents.

## 3. INTENT PREDICTION

In order to predict possible intents given utterances, for each input utterance $U$, we transform it into a vector $y_U$, and then estimate its semantic similarity with vectors for all intents including seen and unseen intents, where vector representations for new intents can be generated from the trained CDSSM by feeding the tri-letter vectors of new intents as the input.

For the utterance $U$, the estimated semantic score of the $k$-th intent is defined as $CosSim(U, I_k)$ in (4). Then predicted intent for each given utterance is decided according to the estimated semantic scores [17, 13].

### 3.1. Unidirectional Estimation

Based on predictive and generative models from Section 2.2.1 and 2.2.2, here for an utterance $U_i$, we define the estimated semantic score of the intent $I_j$ using the predictive model as $S_P(U_i, I_j)$ and using the generative model as $S_G(U_i, I_j)$.

### 3.2. Bidirectional Estimation

Considering that the estimation from two directions may model the similarity in different ways, a bidirectional estimation, $S_{\text{Bi}}(U, I)$, is proposed to incorporate both prediction scores, $S_P(U, I)$ and $S_G(U, I)$, and balance the effectiveness of predictive and generative models:

$$S_{\text{Bi}}(U, I) = \gamma \cdot S_P(U, I) + (1 - \gamma) \cdot S_G(U, I), \quad (9)$$

where $\gamma$ is a weight to control the contributions from both sides.

## 4. EXPERIMENTS

### 4.1. Experimental Setup

The dataset is collected via the Microsoft Cortana conversational agent, where there are more than 100 intents (e.g. get_distance, show_map, change_calendar_entry, etc). The set of intents is segmented into seen and unseen intents to evaluate whether the CDSSM is able to generate proper intent embeddings for improving intent prediction especially for unseen intents. There are a total of 19 different predicates such as find, create, send, get, etc. in all intents. To test the performance of the embedding generation, we randomly chose 7 intents with different predicates as unseen intents, with a total of around 100K utterances. Among arguments of the unseen intents, only 70% words are covered by arguments of seen intents. For the seen intents, there are about 1M annotated utterances, where we use 2/3 for training CDSSM and the rest for testing.

To test the capability of constructing unseen intent embeddings, the CDSSM is trained on the utterances paired with the seen intents. The total number of training iterations is set to 300, the dimension of the convolutional layer is 1000, and the dimension of the semantic layer is 300. The parameter $\gamma$ in (9) is set as 0.5 to allow predictive and generative models contribute equally. To measure the performance of intent prediction, we report the mean average precision at K (MAP@K), where MAP@1 is equal to the prediction accuracy.

### 4.2. Evaluation Results

Experimental results for seen intents and unseen intents are shown in Table 1. CDSSM-Ori only considers the relations between the given utterance and seen intents to determine intents, while CDSSM-Expand additionally considers expanded unseen intent embeddings for prediction.

#### 4.2.1. Effectiveness of Intent Expansion

Before intent expansion, CDSSM-Ori performs from 58% to 68% with various $K$ for seen intents, while it cannot deal with the unseen intents. With the proposed intent expansion, CDSSM-Expand additionally considers new intents, which do not have training samples, and produces similar but slightly worse results as CDSSM-Ori for seen intents. The reason is that considering more intent candidates increases the uncertainty of prediction and may degrade the performance, but the difference is not significant in our experiments. For unseen intents, CDSSM-Expand is able to capture the correct intents and achieve higher than 30% of MAP when $K \geq 3$, which indicates the encouraging performance considering more than 100 intents.

To further analyze the performance of unseen intents, Fig. 3 shows the performance distribution over unseen intents with $K = 1, 3, 5$, where delete_alarm and turn_off_setting perform good and their performance are comparable with seen intents. In addition, there is no seen intent containing the words email and mail, but send_email still shows reasonable performance. The reason is that in the training data, some utterances corresponding to seen intents contain mail related semantics, which can benefit to learning intent embeddings of send_email and result in better performance. On the other hand, get_price_range performs bad, probably because the training data contains few utterances that are related to price and we cannot learn the intent embeddings accurately.

**Table 1**. Intent classification performance on the mean average precision at K (MAP@K) (%).

| Approach | Direction | Seen Intents | | | | | Unseen Intents | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | K=1 | K=3 | K=5 | K=10 | K=30 | K=1 | K=3 | K=5 | K=10 | K=30 |
| CDSSM-Ori | Predictive ($P(I \mid U)$) | 59.00 | 66.29 | 67.47 | 68.30 | 68.77 | - | - | - | - | - |
| | Generative ($P(U \mid I)$) | 45.17 | 52.66 | 54.09 | 55.19 | 55.94 | - | - | - | - | - |
| | Bidirectional | 58.58 | 66.09 | 67.29 | 68.15 | 68.64 | - | - | - | - | - |
| CDSSM-Expand | Predictive ($P(I \mid U)$) | 58.85 | 65.91 | 67.07 | 67.88 | 68.37 | 5.17 | 18.67 | 23.37 | 26.05 | 27.18 |
| | Generative ($P(U \mid I)$) | 44.72 | 52.04 | 53.51 | 54.61 | 55.37 | 6.65 | 23.18 | 26.54 | 28.65 | 29.55 |
| | Bidirectional | 58.31 | 65.60 | 66.80 | 67.67 | 68.17 | 9.07 | 30.99 | 34.52 | 35.98 | 36.58 |

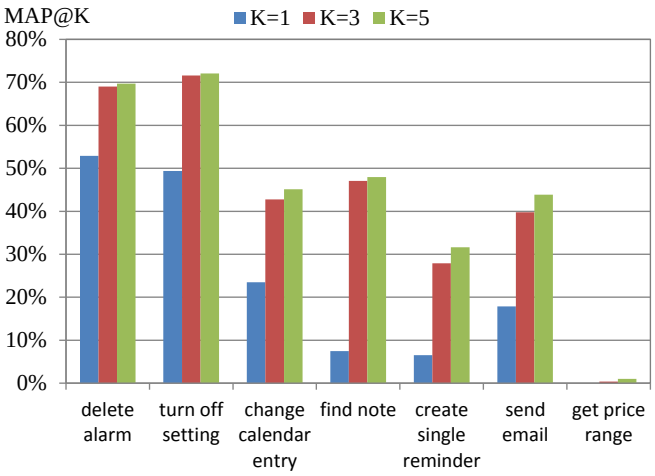


**Fig. 3**. The MAP@K performance distribution over unseen intents.

#### 4.2.2. *Sensitivity to K*

To analyze the quality of top-returned intents, we compare the results using various $K$. For seen intents, both CDSSM-Ori and CDSSM-Expand achieve 58% of MAP when $K = 1$, the performance is better when $K = 3$ (65%-66%), and then continuously increasing $K$ does not show significant improvement. However, for unseen intents, CDSSM-Expand only achieves 9% of MAP when $K = 1$, and $K \geq 3$ gives much better results (higher than 30%). This means that the performance of CDSSM-Expand is more sensitive to the number of returned intents, where the first-returned intent may not accurate enough but the correct intent can still be obtained from the 3-best list. It also motivates the re-ranking approach to further improve the performance in the future.

#### 4.2.3. *Effectiveness of Bidirectional Estimation*

Here we compare the performance among a predictive model, a generative model and a bidirectional model. For seen intents, Table 1 shows that the predictive model ($P(I \mid U)$) is best among three models, and the bidirectional model has similar performance as the predictive model (the difference is not significant). The generative model ($P(U \mid I)$) performs worse in all cases.

However, for unseen intents, the generative model is better than the predictive one, and the bidirectional model has much better performance compared with unidirectional ones. The reason is that the predictive model predicts the intent that maximizes $P(I \mid U)$, where the comparison is across intents (including seen and unseen). Hence, seen intents usually carry higher probabilities from the CDSSM, comparison between seen and unseen intents during prediction may be unfair. In the generative model, the objective maximizes $P(U \mid I)$, where the comparison is across utterances not intents, so seen intents and unseen intents can have fair comparison to achieve better performance. Moreover, the improvement of bidirectional estimation suggests that the predictive model and the generative model can compensate each other, and then provide more robust estimated scores especially for unseen intents, which is crucial to this intent expansion task.

**Table 2**. Intent classification accuracy on seen intents (%).

| Approach | | Accuracy |
|---|---|---|
| Baseline | SVM with `doc2vec` | 45.30 |
| Proposed | CDSSM-Expand: Predictive ($P(I \mid U)$) | **58.85** |
| | CDSSM-Expand: Generative ($P(U \mid I)$) | 44.72 |
| | CDSSM-Expand: Bidirectional | **58.31** |

#### 4.2.4. *Effectiveness of CDSSM*

In addition to the ability of generating more flexible intent embeddings, we plan to evaluate the power of CDSSM features by comparing the performance from other semantic embeddings. We trained paragraph vectors (`doc2vec`) on the corpus [21], where the training set of paragraph vectors is the same as CDSSM takes, the vector dimension is set to 300, and the window size is 3. Then we applied SVM on the trained embeddings for intent prediction [22].

Table 2 shows the performance of different models for seen intents, where `doc2vec` obtains 45% on accuracy, and the predictive model and the bidirectional model perform better than the state-of-the-art baseline, achieving about 58% on accuracy. It shows a promising result and proves the effectiveness of CDSSM features. Note that we use the CDSSM as final decision maker, but it can also be used as a feature extractor as in SVM with `doc2vec`, and could result in better classification performance [15]. We leave such extensions of our approach as part of the future work.

## 5. CONCLUSION

This paper focuses on the task of intent expansion, where a convolutional deep structured semantic model (CDSSM) is applied to perform zero-shot learning of intent embeddings to bridge the semantic relation across domains. The experiments show that CDSSM is capable of generating more flexible intent embeddings without training samples and model re-training, removing the domain constraint in dialogue systems for practical usage. It is also shown that the semantic features carried by CDSSM outperform semantic paragraph vectors for intent classification.

## 6. REFERENCES


[1] Yun-Nung Chen and Alexander Rudnicky, "Dynamically supporting unexplored domains in conversational interactions by enriching semantics with neural word embeddings," in *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 590–595.

[2] Yun-Nung Chen, William Yang Wang, and Alexander I. Rudnicky, "Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2013, pp. 120–125.

[3] Yun-Nung Chen, William Yang Wang, and Alexander I. Rudnicky, "Jointly modeling inter-slot relations by random walk on knowledge graphs for unsupervised spoken language understanding," in *Proceedings of 2015 Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies*. ACL, 2015, pp. 619–629.

[4] Yun-Nung Chen, William Yang Wang, Anatole Gershman, and Alexander I. Rudnicky, "Matrix factorization with knowledge graph propagation for unsupervised spoken language understanding," in *Proceedings of The 53rd Annual Meeting of the Association for Computational Linguistics and The 7th International Joint Conference on Natural Language Processing of the AFNLP*. ACL, 2015.

[5] Mandy Korpusik, Nicole Schmidt, Jennifer Drexler, Scott Cyphers, and James Glass, "Data collection and language understanding of food descriptions," in *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 560–565.

[6] Ali El-Kahky, Xiaohu Liu, Ruhi Sarikaya, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck, "Extending domain coverage of language understanding systems via intent transfer between domains using knowledge graphs and search query click logs," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4067–4071.

[7] Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong, "New transfer learning techniques for disparate label sets," in *Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*. 2015, ACL.

[8] Giuseppe Di Fabbrizio, Gokhan Tur, and Dilek Hakkani-Tür, "Bootstrapping spoken dialog systems with data reuse," in *Proceedings of the 5th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SigDial)*, 2004.

[9] Fabrizio Morbini, Eric Forbell, and Kenji Sagae, "Improving classification-based natural language understanding with non-expert annotation," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SigDial)*, 2014.

[10] Yun-Nung Chen, Dilek Hakkani-Tür, and Gokan Tur, "Deriving local relational surface forms from dependency-based entity embeddings for unsupervised spoken language understanding," in *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 242–247.

[11] Wen-tau Yih, Xiaodong He, and Christopher Meek, "Semantic parsing for single-relation question answering," in *Proceedings of 52nd Annual Meeting of the Annual Meeting of the Association for Computational Linguistics*. 2014, ACL.

[12] Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *Proceedings of the Joint Conference of the 53rd Annual Meeting of the Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the AFNLP*. 2015, ACL Association for Computational Linguistics.

[13] Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He, "Detecting actionable items in meetings by convolutional deep structured semantic models," in *Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 375–382.

[14] Yonatan Belinkov, Mitra Mohtarami, Scott Cyphers, and James Glass, "VectorSLU: A continuous word vector approach to answer selection in community question answering systems," in *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval*, 2015, vol. 15.

[15] Jianfeng Gao, Patrick Pantel, Michael Gamon, Xiaodong He, Li Deng, and Yelong Shen, "Modeling interestingness with deep neural networks," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2014.

[16] Yun-Nung Chen, William Yang Wang, and Alexander I Rudnicky, "Leveraging frame semantics and distributional semantics for unsupervised semantic slot induction in spoken dialogue systems," in *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 584–589.

[17] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Gregoire Mesnil, "A latent semantic model with convolutional-pooling structure for information retrieval," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, 2014, pp. 101–110.

[18] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013, pp. 2333–2338.

[19] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil, "Learning semantic representations using convolutional neural networks for web search," in *Proceedings of the companion publication of the 23rd international conference on World wide web companion*. International World Wide Web Conferences Steering Committee, 2014, pp. 373–374.

[20] Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He, "Learning bidirectional intent embeddings by convolutional deep structred semantic models for spoken language understanding," in *Extended Abstract of The 29th Annual Conference on Neural Information Processing Systems – Machine Learning for Spoken Language Understanding and Interactions Workshop (NIPS-SLU)*, 2015.

[21] Quoc Le and Tomas Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1188–1196.

[22] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, pp. 27, 2011.