Learning Bidirectional Intent Embeddings by Convolutional Deep Structured Semantic Models for Spoken Language Understanding

Yun-Nung Chen[*]
Carnegie Mellon University
yvchen@cs.cmu.edu

Dilek Hakkani-Tür
Microsoft Research
dilek@ieee.org

Xiaodong He
Microsoft Research
xiaohe@microsoft.com

Abstract

The recent surge of intelligent personal assistants motivates spoken language understanding of dialogue systems. Considering high-level semantics, intent embeddings can be viewed as the universal representations that help derive a more flexible intent schema to overcome the domain constraint and the genre mismatch. A convolutional deep structured semantic model (CDSSM) is applied to jointly learn the representations for human intents and associated utterances. Two sets of experiments, intent expansion and actionable item detection, are conducted to evaluate the power of the learned intent embeddings. The representations bridge the semantic relation between seen and unseen intents for intent expansion, and connect intents from different genres for actionable item detection. The discussion and analysis of experiments provide a future direction for reducing human effort of data annotation and eliminating domain and genre constraints for spoken language understanding.

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, Facebook M, etc), where spoken language understanding (SLU) is a key component of a dialogue system, which parses user utterances into corresponding intents and associated semantic slots. Typically all domains are implemented independently, where training intent detectors and slot taggers require domain-specific manually annotated data. However, the intents are usually predefined and inflexible to expand and transfer cross domains. 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. A standard solution is to re-design a semantic schema adding new intents with associated slots to cover the new intents, which requires human effort for annotation and model re-training. These issues remain the biggest challenge for SLU [1, 2].

Similarly, genre mismatch is another challenge of SLU. With task-oriented dialogue systems (human-machine genre), such as personal assistants that can schedule meetings and send emails to aid users, how can a system transfer the functionality to human dialogues (human-human genre)? Most of the previous work on language understanding of human-human conversations focused on analyzing task-oriented dialogues such as in customer care centers, and aimed to infer semantic representations and bootstrap language understanding models [3, 4, 5, 6, 7]. These would then be used in human-machine dialogue systems that automate the targeted task, such as travel arrangements. This work takes the reversed direction, which transfers intents from the human-machine genre to the human-human genre.

[*]The work was done during the summer internship at Microsoft Research.

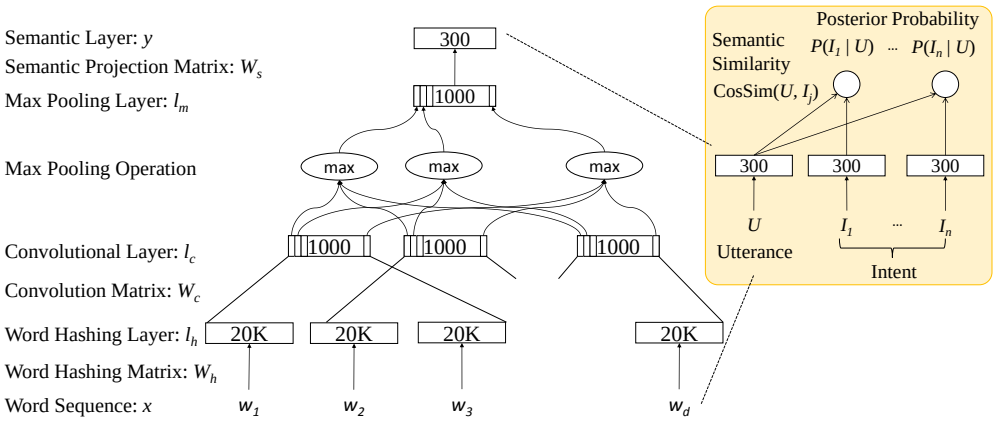

Figure 1: Illustration of the CDSSM architecture for the predictive model.

With recent works on word embeddings and paragraph embeddings [8, 9], this paper addresses the issue about domain and genre constraints by learning intent embeddings to obtain high-level semantic representations for two tasks, intent expansion and actionable item detection. Intent expansion is a task of predicting the unseen intents based on the data with seen intents, while actionable item detection is to detect intents in human-human conversations based on human-machine interactions. Both tasks address the issue about cross-domain constraints; the first task bridges the semantics from different domains, and the second task connects the relation between two genres (human-human genre and human-machine genre).

Previous work bootstrapped SLU models for a new domain by re-using the annotated intent data from other domains [10, 11]. Kim et al. also proposed to automatically generate the mapping between semantic slots across domains to connect information cross domains [2]. Both studies implied that semantics from different domains can be shared and the connection may help domain adaptation and expansion. Instead of modeling the relations between intents from different sources, 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 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 the unseen intent representations based on the trained model, which benefits by the semantics from other domains. Then we can expand and adapt intents to different domains and genres without human annotation and model retraining.

In this paper, we treat intent modeling as an utterance classification task, where each user utterance corresponds to an intent or an action. Recent studies used CDSSM to map questions into relation-entity triples for question answering [12, 13], which motivates us to use CDSSM for capturing relations from intent-utterance pairs [14], 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 [15, 16, 17], the representations of intents and utterances can incorporate more informative cues from large data. Hence, this paper focuses on taking CDSSM features to help model intents and conducts experiments for different tasks to prove the capability of intent embeddings produced by CDSSM.

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. 1 [15, 18, 19]. 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 [19]. 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. 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 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:

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

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),

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

The convolutional and max-pooling layers are able to capture prominent words of the word sequences [15, 18]. As illustrated in Fig. 1, 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.

$$y = \tanh(W_s^T l_m), \tag{3}$$

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

2.2 Training Procedure

The meeting data contains utterances and associated actions. The idea of this model is to learn the embeddings for utterances and intents such that utterances with the same intents can be close to each other in the continuous space. We define a semantic score between an utterance U and an action I using the cosine similarity between their embeddings, y_U and y_I, as $\mathrm{CosSim}(U, I)$.

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, and a model can be trained by maximizing the likelihood of the correctly associated intents given training utterances.

$$P(I \mid U) = \frac{\exp(\mathrm{CosSim}(U, I))}{\sum_{I'} \exp(\mathrm{CosSim}(U, I'))}, \ \Lambda(\theta_1) = \log \prod_{(U, I^+)} P(I^+ \mid U) \tag{4}$$

where I' is an intent candidate. $\theta_1 = \{W_c, W_s\}$ are the parameters of the model, which is optimized using mini-batch stochastic gradient descent (SGD) [19].

2.2.2 Generative Model

Similarly, we can estimate the posterior probability of an utterance given an intent using the reverse setting to obtain $P(U \mid I)$, which is the generative model that emits the utterances for each intent. Also, the parameters of the model θ_2 are optimized similarly and performs a reversed estimation for the relation between utterances and intents.

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

Approach			Seen Intents					Unseen Intents				
			K=1	K=3	K=5	K=10	K=30	K=1	K=3	K=5	K=10	K=30
Ori.	(a)	$P(I\|U)$	59.0	66.3	67.5	68.3	68.8	-	-	-	-	-
	(b)	$P(U\|I)$	45.2	52.7	54.1	55.2	56.0	-	-	-	-	-
	(c)	Bidir	58.6	66.1	67.3	68.2	68.6	-	-	-	-	-
Exp.	(d)	$P(I\|U)$	58.9	65.9	67.1	67.9	68.4	5.2	18.7	23.4	26.1	27.2
	(e)	$P(U\|I)$	44.7	52.0	53.5	54.6	55.4	6.7	23.2	26.5	28.7	29.6
	(f)	Bidir	58.3	65.6	66.8	67.7	68.2	9.1	31.0	34.5	36.0	36.6

2.3 Score 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 $S_G(U_i, I_j)$.

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:

$$S_{\text{Bi}}(U, I) = \gamma \cdot S_P(U, I) + (1 - \gamma) \cdot S_G(U, I), \tag{5}$$

where γ is a weight to control the contributions from both sides.

3 Unseen Intent Prediction Experiments

In order to predict the possible intents given utterances, for each utterance U, we transform it into a vector y_U, and then estimate the semantic similarity with vectors for all intents including seen and unseen intents, where the vector representations for new intents can be generated from the trained CDSSM by feeding the tri-letter vectors of the new intent as input. For the utterance U, the estimated semantic score of the k-th intent is defined as $S(U, I_k)$. Then predicted intent for each given utterance is decided according to the estimated semantic scores [18, 14]: $I^* = \arg\max_k S(U, I_k)$.

3.1 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 total 19 different verbs such as find, create, send, get, etc. in all intents. To test the performance of the embedding generation, we chose 7 intents with different verbs as unseen intents, with in total around 100K utterances. 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 γ in (5) 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.

3.2 Evaluation Results

Experimental results for seen intents and unseen intents are shown in Table 1. The original models (rows (a)-(c)) only consider the relations between the given utterance and seen intents to determine intents, while the expanded models (rows (d)-(f)) additionally consider expanded unseen intent embeddings for prediction. The results before intent expansion achieve from 58% to 68% of MAP@K with various K for seen intents, while it cannot deal with the unseen intents. With the proposed intent expansion, the expanded models additionally consider new intents without training samples,

and produces similar but slightly worse results as original models 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, expanded models are 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.

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 intents). 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. To summarize, the experiments show that the learned embeddings capture the semantics borrowed from other domains and can be used to flexibly expand the intents through high-level representations.

4 Actionable Item Detection Experiments

This task investigates actionable item detection in meetings (human-human genre), where the intelligent assistant dynamically provides the participants access to information (e.g. scheduling a meeting, taking notes) without interrupting the meetings. A CDSSM is applied to learn the latent semantics for human actions and utterances from human-machine and human-human interactions. In order to predict the possible actions given meeting utterances, for each utterance U, we transform it into a vector y_U, and then estimate the semantic similarity with vectors for all intended actions I. The estimated semantic scores can be used in two ways [18]: 1) as prediction scores of the actionable item detector, and 2) as features inputted to a multi-class classifier and then convert to the final estimation outputted by the classifier.

4.1 Experimental Setup

The dataset is from the ICSI meeting corpus [20], where 22 meetings previously used as test and dev sets are included for the actionable item detection task [21]. These include three types of meetings, Bed, Bmr, and Bro, which include regular project discussions between colleagues and conversations between students and their advisors. The total numbers of utterances are 4544, 9227, and 7264 for Bed, Bmr, and Bro respectively. Actionable items were manually annotated, where the annotation schema was designed based on the Microsoft Cortana conversational agent schema. We identified 10 actions that are relevant to meeting scenarios: find_calendar_entry, create_calendar_entry, open_agenda, add_agenda_item, create_single_reminder, make_call, search, send_email, find_email, and open_setting. There are total 318 utterances annotated with actionable items, which accounts for about 2% of all utterances.

To compare with the matched data for CDSSM, we perform the experiments using Mismatch-CDSSM and Match-CDSSM, where Mismatch-CDSSM is a CDSSM trained on conversational agent data, which mismatches with the target genre, and Match-CDSSM is a CDSSM trained on meeting data, which matches with the target genre. The conversational agent data is the same as the prediction task. For meeting data, we conduct the experiments on the manual transcripts using the same CDSSM setting. To test the generalization to different meeting types, we take one of meeting types as training data and test on each of remaining two. Hence, we have 6 sets of experiments and report the average of AUC scores for evaluation, which is similar to 6-fold cross-validation. The multi-class classifier we apply for actionable item detection is the SVM with RBF kernel using a default setting [22]. The parameter γ in (5) is set as 0.5 to allow predictive and generative models contribute equally. Due to imbalanced classes (number of non-actionable utterances is larger than

Table 2: Actionable item detection on average area of the precision-recall curve (AUC) (%).

Approach			#dim	Mismatch-CDSSM			Match-CDSSM		
				$P(A\|U)$	$P(U\|A)$	Bidir	$P(A\|U)$	$P(U\|A)$	Bidir
(a)	Sim ($\mathrm{CosSim}(U, A)$)			47.45	48.17	49.10	56.33	43.39	50.57
(b)	SVM	Embeddings	300	53.07	48.07	55.71	64.33	65.58	**69.27**
(c)		(b) + Sim	311	52.80	54.95	**59.09**	64.52	64.81	68.86

number of actionable ones), the evaluation focuses on detection performance for each action. Here for each action, we use the area under the precision-recall curve (AUC) as the metric to evaluate whether the detector is able to effectively detect it for test utterances. In the experiments, we report the average AUC scores over all classes (10 actions plus others).

4.2 Evaluation Results

Experimental results with different CDSSMs are shown in Table 2. Row (a) uses the semantic similarity as final prediction scores, while rows (b)-(c) use the similarity as features and then train an SVM to estimate the final prediction scores, where row (b) takes utterance embedding vectors as features, and row (c) includes the semantic similarity as additional features for the classifier. Hence the dimension of features is 311, including 300 values of utterance embeddings and 11 similarity scores for 10 actions and others.

Treating the semantic similarity as final prediction scores achieves 49.10% and 50.57% for Mismatch-CDSSM and Match-CDSSM respectively. Considering that the learned embeddings do not fit the target genre well, the similarity treated as features of a classifier can be combined with other features to automatically adapt the reliability of the similarity features. Row (b) shows the improved performance using only utterance embeddings (from 49.10% to 55.71% for Mismatch-CDSSM and from 50.57% to 69.27% for Match-CDSSM). Including the similarity scores as additional features can further improve the performance for Mismatch-CDSSM (from 55.71% to 59.09%); however, for Match-CDSSM, additionally adding similarity scores as features does not show the improvement, since the data for training CDSSM is from meetings, and the performance cannot be improved when there is no genre mismatch. In addition, Mismatch-CDSSM is able to capture the semantics of the unseen intents that share the semantics from other seen intents. For example, send_email does not appear in the training data, but we can still learn its embeddins from send_message and find_email, showing the power of CDSSM.

From Table 2, it is shown that all results from the bidirectional estimation significantly outperform the results using unidirectional estimation across all CDSSMs and all methods except for rows (a) from Match-CDSSM. Comparing between the predictive model ($P(A \mid U)$) and the generative model ($P(U \mid A)$), the performance is similar and does not show that a certain direction is better in most cases. 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 for the goal of actionable item detection. In sum, the experiments show that the learned intent embeddings carry the crucial high-level semantics and can be applied to different genres for easy adaptation and extension.

5 Conclusions

This paper focuses on learning intent embeddings for the tasks of intent expansion and actionable item detection, where a convolutional deep structured semantic model (CDSSM) is applied to perform representation learning to bridge the semantic relation across domains and across genres. The experiments show that CDSSM is capable of generating more flexible intent embeddings to remove the domain constraint in dialogue systems for intent expansion. Also the intent embeddings learned from the human-machine genre can be applied to detect actionable utterances in the human-human genre, showing the robustness to genre mismatch. In sum, the paper highlights a future research direction for bridging semantics across domains and genres.

References

[1] 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 *Proceedings of ICASSP*, 2014.

[2] Young-Bum Kim, Karl Stratos, Ruhi Sarikaya, and Minwoo Jeong. New transfer learning techniques for disparate label sets. In *Proceedings ACL*, 2015.

[3] Gokhan Tur, Andreas Stolcke, L Lynn Voss, John Dowding, Benoît Favre, Raquel Fernández, Matthew Frampton, Michael W Frandsen, Clint Frederickson, Martin Graciarena, et al. The CALO meeting speech recognition and understanding system. In *Proceedings of SLT*, 2008.

[4] 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 *Proceedings of ASRU*, 2013.

[5] 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 ACL-IJCNLP*, 2015.

[6] 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 NAACL-HLT*, 2015.

[7] Yun-Nung Chen, Ming Sun, Alexander I. Rudnicky, and Anatole Gershman. Leveraging behavioral patterns of mobile applications for personalized spoken language understanding. In *Proceedings ICMI*, 2015.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, 2013.

[9] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of ICML*, 2014.

[10] Giuseppe Di Fabbrizio, Gokhan Tur, and Dilek Hakkani-Tür. Bootstrapping spoken dialog systems with data reuse. In *Proceedings of SigDial*, 2004.

[11] Fabrizio Morbini, Eric Forbell, and Kenji Sagae. Improving classification-based natural language understanding with non-expert annotation. In *Proceedings of SigDial*, 2014.

[12] Wen-tau Yih, Xiaodong He, and Christopher Meek. Semantic parsing for single-relation question answering. In *Proceedings of ACL*, 2014.

[13] 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 ACL-IJCNLP*, 2015.

[14] Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He. Detecting actionable items in meetings by convolutional deep structured semantic models. In *Proceedings of ASRU*, 2015.

[15] Jianfeng Gao, Patrick Pantel, Michael Gamon, Xiaodong He, Li Deng, and Yelong Shen. Modeling interestingness with deep neural networks. In *Proceedings of EMNLP*, 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 *Proceedings of SLT*, 2014.

[17] Yun-Nung Chen and Alexander Rudnicky. Dynamically supporting unexplored domains in conversational interactions by enriching semantics with neural word embeddings. In *Proceedings of SLT*, 2014.

[18] 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 CIKM*, 2014.

[19] 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 CIKM*, 2013.

[20] Adam Janin, Don Baron, Jane Edwards, Dan Ellis, Davis Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, Andreas Stolcke, and Chuck Wooters. The ICSI meeting corpus. In *Proceedings of ICASSP*, 2003.

[21] Jeremy Ang, Yang Liu, and Elizabeth Shriberg. Automatic dialog act segmentation and classification in multiparty meetings. In *Proceedings of ICASSP*, 2005.

[22] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.