Modeling Speech Acts in Asynchronous Conversations: A Neural-CRF Approach

Shafiq Joty
Nanyang Technological University
School of Computer Science
and Engineering
srjoty@ntu.edu.sg

Tasnim Mohiuddin
Nanyang Technological University
School of Computer Science
and Engineering
mohi0004@e.ntu.edu.sg

Participants in an asynchronous conversation (e.g., forum, e-mail) interact with each other at different times, performing certain communicative acts, called speech acts (e.g., question, request). In this article, we propose a hybrid approach to speech act recognition in asynchronous conversations. Our approach works in two main steps: a long short-term memory recurrent neural network (LSTM-RNN) first encodes each sentence separately into a task-specific distributed representation, and this is then used in a conditional random field (CRF) model to capture the conversational dependencies between sentences. The LSTM-RNN model uses pretrained word embeddings learned from a large conversational corpus and is trained to classify sentences into speech act types. The CRF model can consider arbitrary graph structures to model conversational dependencies in an asynchronous conversation. In addition, to mitigate the problem of limited annotated data in the asynchronous domains, we adapt the LSTM-RNN model to learn from synchronous conversations (e.g., meetings), using domain adversarial training of neural networks. Empirical evaluation shows the effectiveness of our approach over existing ones: (i) LSTM-RNNs provide better task-specific representations, (ii) conversational word embeddings benefit the LSTM-RNNs more than the off-the-shelf ones, (iii) adversarial training gives better domain-invariant representations, and (iv) the global CRF model improves over local models.

1. Introduction

With the advent of Internet technologies, communication media like e-mails and discussion forums have become commonplace for discussing work, issues, events, and experiences. Participants in these media interact with each other **asynchronously** by

Submission received: 15 October 2017; revised version received: 5 June 2018; accepted for publication: 22 August 2018.

doi:10.1162/coli_a_00339

writing at different times. This generates a type of conversational discourse, where information flow is often not sequential as in monologue (e.g., news articles) or in synchronous conversation (e.g., instant messaging). As a result, discourse structures such as topic structure, coherence structure, and conversational structure in these conversations exhibit different properties from what we observe in monologue or in synchronous conversation (Joty, Carenini, and Ng 2013; Louis and Cohen 2015).

Participants in an asynchronous conversation interact with each other in complex ways, performing certain communicative acts like asking questions, requesting information, or suggesting something. These are called **speech acts** (Austin 1962). For example, consider the excerpt of a forum conversation[1] from our corpus in Figure 1. The participant who posted the first comment, $\mathbf{C_1}$, describes his situation in the first two sentences, and then asks a *question* in the third sentence. Other participants respond to the query by *suggesting* something or *asking* for clarification. In this process, the participants get into a conversation by taking turns, each of which consists of one or more speech acts. The two-part structures across posts like question-answer and request-grant are called **adjacency pairs** (Schegloff 1968).

Identification of speech acts is an important step toward deep conversational analysis (Bangalore, Di Fabbrizio, and Stent 2006), and has been shown to be useful in many downstream applications, including summarization (Murray et al. 2006; McKeown, Shrestha, and Rambow 2007), question answering (Hong and Davison 2009), collaborative task learning agents (Allen et al. 2007), artificial companions for people to use the Internet (Wilks 2006), and flirtation detection in speed-dates (Ranganath, Jurafsky, and McFarland 2009).

Availability of large annotated corpora like the Meeting Recorder Dialog Act (MRDA) (Dhillon et al. 2004) or the Switchboard-DAMSL (SWBD) (Jurafsky, Shriberg, and Biasca 1997) corpus has fostered research in data-driven automatic speech act recognition in synchronous domains like meeting and phone conversations (Ries 1999; Stolcke et al. 2000; Dielmann and Renals 2008).[2] However, such large corpora are not available in the asynchronous domains, and many of the existing (small-sized) corpora use task-specific speech act tagsets (Cohen, Carvalho, and Mitchelle 2004; Ravi and Kim 2007; Bhatia, Biyani, and Mitra 2014) as opposed to a standard one. The unavailability of large annotated data sets with standard tagsets is one of the reasons for speech act recognition not getting much attention in asynchronous domains.

Previous attempts in automatic (sentence-level) speech act recognition in asynchronous conversations (Jeong, Lin, and Lee 2009; Qadir and Riloff 2011; Tavafi et al. 2013; Oya and Carenini 2014) suffer from at least one of the following two technical limitations.

First, they use a bag-of-words (BOW) *representation* (e.g., unigram, bigram) to encode lexical information of a sentence. However, consider the Suggestion sentences in the example. Arguably, a model needs to consider the structure (e.g., word order) and the compositionality of phrases to identify the right speech act for an utterance. Furthermore, BOW representation could be quite sparse, and may not generalize well when used in classification models. Recent research suggests that a condensed distributed representation learned by a neural model on the target task (e.g., speech act classification) is more effective. The task-specific training can be further improved by pretrained word embeddings (Goodfellow, Bengio, and Courville 2016).

1 Taken from `http://www.qatarliving.com/forum/advice-help/posts/study-canada`.
2 Speech acts are also known as "dialog acts" in the literature.

Second, existing approaches mostly disregard *conversational dependencies* between sentences inside a comment and across comments. For instance, consider the example in Figure 1 again. The Suggestions are answers to Questions asked in a previous comment. We therefore hypothesize that modeling inter-sentence relations is crucial for speech act recognition. We have tagged the sentences in Figure 1 with human annotations (HUMAN) and with the predictions of a local (LOCAL) classifier that considers word order for sentence representation but classifies each sentence separately or individually. Prediction errors are underlined and highlighted in red. Notice the first and second sentences of comment $\mathbf{C_4}$, which are mistakenly tagged as Statement and Response, respectively, by our best local classifier. We hypothesize that some of the errors made by the local classifier could be corrected by utilizing a global joint model that is trained to perform a collective classification, taking into account the conversational dependencies between sentences (e.g., adjacency relations like Question-Suggestion).

C_1: My son wish to do his bachelor degree in Mechanical Engineering in an affordable Canadian university.
⇒ HUMAN: Statement, LOCAL: Statement, GLOBAL: Statement

The information available in the net and the people who wish to offer services are too many and some are misleading.
⇒ HUMAN: Statement, LOCAL: Statement, GLOBAL: Statement

The preliminary preparations, eligibility, the require funds etc., are some of the issues which I wish to know from any panel members of this forum who is aware and had gone through similar procedures to obtain an admission in an university abroad.
⇒ HUMAN: Question, LOCAL: Statement, GLOBAL: Statement

C_3: [truncated] take a list of canadian universities and then create a table and insert all the relevant information by reading each and every program info on the web.
⇒ HUMAN: Suggestion, LOCAL: Suggestion, GLOBAL: Suggestion

Without doing a research my advice would be to apply to UVIC... for the following reasons... 1. good egineering school, 2 affordable, 3 strong co-op, 4. beautiful and safe city.
⇒ HUMAN: Suggestion, LOCAL: Suggestion, GLOBAL: Suggestion

UBC is good too... but it is expensive particularly for international students due to tuition differential.... and pls pls pls.. dont waste your money on intermediaries or so called consultants... do it yourself.. most of them accept on-line application or email application.
⇒ HUMAN: Suggestion, LOCAL: Suggestion, GLOBAL: Suggestion

most of them accept on-line or email application.
⇒ HUMAN: Statement, LOCAL: Statement, GLOBAL: Statement

Good luck !!
⇒ HUMAN: Polite, LOCAL: Polite, GLOBAL: Polite

C_4: snakyy21: UVIC is a short form of? I have already started researching for my brother and found "College of North Atlantic" and planning to visit their branch in Qatar to inquire about more details
⇒ HUMAN: Question, LOCAL: Statement, GLOBAL: Question

but not sure about the reputation..
⇒ HUMAN: Statement, LOCAL: Response, GLOBAL: Statement

C_5: thank you for sharing useful tips will follow your advise.
⇒ HUMAN: Polite, LOCAL: Polite, GLOBAL: Polite

Figure 1
Example of a forum conversation (truncated) with HUMAN annotations and automatic predictions by a LOCAL classifier and a GLOBAL classifier for speech acts (e.g., Statement, Suggestion). The incorrect decisions are underlined and marked with red color.

However, unlike synchronous conversations (e.g., meeting, phone), modeling conversational dependencies between sentences in an asynchronous conversation is challenging, especially when the thread structure (e.g., "reply-to" links between comments) is missing, which is also our case. The conversational flow often lacks sequential dependencies in its temporal/chronological order. For example, if we arrange the sentences as they arrive in the conversation, it becomes hard to capture any dependency between the act types because the two components of the adjacency pairs can be far apart in the sequence. This leaves us with one open research question: How do we model the dependencies between sentences in a single comment and between sentences across different comments? In this article, we attempt to address this question by designing and experimenting with conditional structured models over arbitrary graph structures of the conversation. Apart from the underlying discourse structure (sequence vs. graph), asynchronous conversations differ from synchronous conversations in *style* (spoken vs. written) and in *vocabulary* usage (meeting conversations on some focused topics vs. conversations on any topic of interest in a public forum). In this article, we propose to use domain adaptation methods in the neural network framework to model these differences in the sentence encoding process.

More concretely, we make the following contributions in speech act recognition for asynchronous conversations. First, we propose to use a recurrent neural network (RNN) with a long short-term memory (LSTM) hidden layer to compose phrases in a sentence and to represent the sentence using distributed condensed vectors (i.e., embeddings). These embeddings are trained directly on the speech act classification task. We experiment with both unidirectional and bidirectional RNNs. Second, we train (task-agnostic) word embeddings from a large conversational corpus, and use it to boost the performance of the LSTM-RNN model. Third, we propose conditional structured models in the form of pairwise conditional random fields (CRF) (Murphy 2012) over arbitrary conversational structures. We experiment with different variations of this model to capture different types of interactions between sentences inside the comments and across the comments in a conversational thread. These models use the LSTM-encoded vectors as feature vectors for learning to classify sentences in a conversation collectively.

Furthermore, to address the problem of insufficient training data in the asynchronous domains, we propose to use the available labeled data from synchronous domains (e.g., meetings). To make the best use of this out-of-domain data, we adapt our LSTM-RNN encoder to learn task-specific sentence representations by modeling the differences in style and vocabulary usage between the two domains. We achieve this by using the recently proposed **domain adversarial** training methods of neural networks (Ganin et al. 2016). As a secondary contribution, we also present and release a forum data set annotated with a standard speech act tagset.

We train our models in various settings with synchronous and asynchronous corpora, and we evaluate on one synchronous meeting data set and three asynchronous data sets—two forum data sets and one e-mail data set. We also experimented with different pretrained word embeddings in the LSTM-RNN model. Our main findings are: (i) LSTM-RNNs provide better sentence representation than BOW and other unsupervised methods; (ii) bidirectional LSTM-RNNs, which encode a sentence using two vectors, provide better representation than the unidirectional ones; (iii) word embeddings pretrained on a large conversational corpus yield significant improvements; (iv) the globally normalized joint models (CRFs) improve over local models for certain graph structures; and (v) domain adversarial training improves the results by inducing domain-invariant features. The source code, the pretrained word embeddings,

and the new data sets are available at https://ntunlpsg.github.io/demo/project/speech-act/.

After discussing related work in Section 2, we present our speech act recognition framework in Section 3. In Section 4, we present the data sets used in our experiments along with our newly created corpus. The experiments and analysis of results are presented in Section 5. Finally, we summarize our contributions with future directions in Section 6.

2. Related Work

Three lines of research are related to our work: (i) compositionality with LSTM-RNNs, (ii) conditional structured models, and (iii) speech act recognition in asynchronous conversations.

2.1 LSTM-RNNs for Composition

RNNs are arguably the most popular deep learning models in natural language processing, where they have been used for both encoding and decoding a text—for example, language modeling (Mikolov 2012a; Tran, Zukerman, and Haffari 2016), machine translation (Bahdanau, Cho, and Bengio 2015), summarization (Rush, Chopra, and Weston 2015), and syntactic parsing (Dyer et al. 2015). RNNs have also been used as a sequence tagger, as in opinion mining (Irsoy and Cardie 2014; Liu, Joty, and Meng 2015), named entity recognition (Lample et al. 2016), and part-of-speech tagging (Plank, Søgaard, and Goldberg 2016).

Relevant to our implementation, Kalchbrenner and Blunsom (2013) use a simple RNN to model sequential dependencies between act types for speech act recognition in phone conversations. They use a convolutional neural network (CNN) to compose sentence representations from word vectors. Lee and Dernoncourt (2016) use a similar model, but they also experiment with RNNs to compose sentence representations. Similarly, Khanpour, Guntakandla, and Nielsen (2016) use an LSTM-based RNN to compose sentence representations. Ji, Haffari, and Eisenstein (2016) propose a latent variable RNN that can jointly model sequences of words (i.e., language modeling) and discourse relations between adjacent sentences. The discourse relations are modeled with a latent variable that can be marginalized during testing. In one experiment, they use coherence relations from the Penn Discourse Treebank corpus as the discourse relations. In another setting, they use speech acts from the SWBD corpus as the discourse relations. They show improvements on both language modeling and discourse relation prediction tasks. Shen and Lee (2016) use an attention-based LSTM-RNN model for speech act classification. The purpose of the attention is to focus on the relevant part of the input sentence. Tran, Zukerman, and Haffari (2017) use an online inference technique similar to the forward pass of the traditional forward-backward inference algorithm to improve upon the greedy decoding methods typically used in the RNN-based sequence labeling models. Vinyals and Le (2015) and Serban et al. (2016) use RNN-based encoder-decoder framework for conversation modeling. Vinyals and Le (2015) use a single RNN to encode all the previous utterances (i.e., by concatenating the tokens of previous utterances), whereas Serban et al. (2016) use a hierarchical encoder—one to encode the words in each utterance, and another to connect the encoded context vectors.

Li et al. (2015) compare recurrent neural models with recursive (syntax-based) models for several NLP tasks and conclude that recurrent models perform on par with the recursive for most tasks (or even better). For example, recurrent models outperform

recursive on sentence level sentiment classification. This finding motivated us to use recurrent models rather than recursive ones.

2.2 Conditional Structured Models

There has been an explosion of interest in CRFs for solving structured output problems in NLP; see Smith (2011) for an overview. The most common type of CRF has a linear chain structure that has been used in sequence labeling tasks like part-of-speech (POS) tagging, chunking, named entity recognition, and many others (Sutton and McCallum 2012). Tree-structured CRFs have been used for parsing (e.g., Finkel, Kleeman, and Manning 2008).

The idea of combining neural networks with graphical models for speech act recognition goes back to Ries (1999), in which a feed-forward neural network is used to model the emission distribution of a supervised hidden Markov model (HMM). In this approach, each input sentence in a dialogue sequence is represented as a BOW vector, which is fed to the neural network. The corresponding sequence of speech acts is given by the hidden states of the HMM. Surendran and Levow (2006) first use support vector machines (SVMs) (i.e., local classifier) to estimate the probability of different speech acts for each individual utterance by combining sparse textual features (i.e., bag of *n*-grams) and dense acoustic features. The estimated probabilities are then used in the Viterbi algorithm to find the most probable tag sequence for a conversation. Julia and Iftekharuddin (2008) use a fusion of SVM and HMM classifiers with textual and acoustic features to classify utterances into speech acts.

More recently, Lample et al. (2016) proposed an LSTM-CRF model for named entity recognition (NER), which first generates a bi-directional LSTM encoding for each input word, and then it passes this representation to a CRF layer, whose task is to encourage global consistency of the NER tags. For each input word, the input to the LSTM consists of a concatenation of the corresponding word embedding and of character-level bi-LSTM embeddings for the current word. The whole network is trained end-to-end with backpropagation, which can be done effectively for chain-structured graphs. Ma and Hovy (2016) proposed a similar framework, but they replace the character-level bi-LSTM with a CNN. They evaluated their approach on POS and NER tagging tasks. Strubell et al. (2017) extended these models by substituting the word-level LSTM with an *iterated dilated* convolutional neural network, a variant of CNN, for which the effective context window in the input can grow exponentially with the depth of the network, while having a modest number of parameters to estimate. Their approach permits fixed-depth convolutions to run in parallel across entire documents, thus making use of GPUs, which yields up to 20-fold speed up, while retaining performance comparable to that of LSTM-CRF. Speech act recognition in asynchronous conversation posits a different problem, where the challenge is to model arbitrary conversational structures. In this work, we propose a general class of models based on pairwise CRFs that work on arbitrary graph structures.

2.3 Speech Act Recognition in Asynchronous Conversation

Previous studies on speech act recognition in asynchronous conversation have used supervised, semi-supervised, and unsupervised methods.

Cohen, Carvalho, and Mitchell (2004) first use the term *e-mail speech act* for classifying e-mails based on their acts (e.g., deliver, meeting). Their classifiers do not capture any contextual dependencies between the acts. To model contextual dependencies,

Carvalho and Cohen (2005) use a collective classification approach with two different classifiers, one for content and one for context, in an iterative algorithm. The content classifier only looks at the content of the message, whereas the context classifier takes into account both the content of the message and the dialog act labels of its parent and children in the thread structure of the e-mail conversation. Our approach is similar in spirit to their approach with three crucial differences: (i) our CRFs are globally normalized to surmount the *label bias* problem, while their classifiers are normalized locally; (ii) the graph structure of the conversation is given in their case, which is not the case with ours; and (iii) their approach works at the comment level, whereas we work at the sentence level.

Identification of adjacency pairs like *question-answer* pairs in e-mail discussions using supervised methods was investigated in Shrestha and McKeown (2004) and Ravi and Kim (2007). Ferschke, Gurevych, and Chebotar (2012) use speech acts to analyze the collaborative process of editing Wiki pages, and apply supervised models to identify the speech acts in Wikipedia Talk pages. Other sentence-level approaches use supervised classifiers and sequence taggers (Qadir and Riloff 2011; Tavafi et al. 2013; Oya and Carenini 2014). Vosoughi and Roy (2016) trained off-the-shelf classifiers (e.g., SVM, naive Bayes, Logistic Regression) with syntactic (e.g., punctuations, dependency relations, abbreviations) and semantic feature sets (e.g., opinion words, vulgar words, emoticons) to classify tweets into six Twitter-specific speech act categories.

Several semi-supervised methods have been proposed for speech act recognition in asynchronous conversation. Jeong, Lin, and Lee (2009) use semi-supervised boosting to tag the sentences in e-mail and forum discussions with speech acts by inducing knowledge from annotated spoken conversations (MRDA meeting and SWBD telephone conversations). Given a sentence represented as a set of trees (i.e., dependency, *n*-gram tree, and POS tag tree), the boosting algorithm iteratively learns the best feature set (i.e., sub-trees) that minimizes the errors in the training data. This approach does not consider the dependencies between the act types, something we successfully exploit in our work. Zhang, Gao, and Li (2012) also use semi-supervised methods for speech act recognition in Twitter. They use a transductive SVM and a graph-based label propagation framework to leverage the knowledge from abundant unlabeled data. In our work, we leverage labeled data from synchronous conversations while adapting our model to account for the shift in the data distributions of the two domains. In our unsupervised adaptation scenario, we do not use any labeled data from the target (asynchronous) domain, whereas in the semi-supervised scenario, we use some labeled data from the target domain.

Among methods that use unsupervised learning, Ritter, Cherry, and Dolan (2010) propose two HMM-based unsupervised conversational models for modeling speech acts in Twitter. In particular, they use a simple HMM and a HMM+Topic model to cluster the Twitter posts (not the sentences) into act types. Because they use a unigram language model to define the emission distribution, their simple HMM model tends to find some topical clusters in addition to the clusters that are based on speech acts. The HMM+Topic model tries to separate the act indicators from the topic words. By visualizing the type of conversations found by the two models, they show that the output of the HMM+Topic model is more interpretable than that of the HMM one; however, their classification accuracy is not empirically evaluated. Therefore, it is not clear whether these models are actually useful, and which of the two models is a better speech act tagger. Paul (2012) proposes using a mixed membership Markov model to cluster sentences based on their speech acts, and show that this model outperforms a simple HMM. Joty, Carenini, and Lin (2011) propose unsupervised models for speech

act recognition in e-mail and forum conversations. They propose a HMM+Mix model to separate out the topic indicators. By training their model based on a conversational structure, they demonstrate that conversational structure is crucial to learning a better speech act recognition model. In our work, we also demonstrate that conversational structure is important for modeling conversational dependencies, however, we do not use any given structure; rather, we build models based on arbitrary graph structures.

3. Our Approach

Let $\mathbf{s}_m^n$ denote the m-th sentence of comment n in an asynchronous conversation; our goal is to find the corresponding speech act tag $y_m^n \in \mathcal{T}$, where $\mathcal{T}$ is the set of available tags. Our approach works in two main steps, as outlined in Figure 2. First, we use a RNN to encode each sentence into a task-specific distributed representation (i.e., embedding) by composing the words sequentially. The RNN is trained to classify sentences into speech act types, and is adapted to give *domain-invariant* sentence features when trained to leverage additional data from synchronous domains (e.g., meetings). In the second step, a structured model takes the sentence embeddings as input, and defines a joint distribution over sentences to capture the conversational dependencies. In the following sections, we describe these steps in detail.

3.1 Learning Task-Specific Sentence Representation

One of our main hypotheses is that a sentence representation method should consider the word order of the sentence. To this end, we use a RNN to encode each sentence into a vector by processing its words sequentially, at each time step combining the current input with the previous hidden state. Figure 3(a) demonstrates the process for three sentences. Initially, we create an embedding matrix $E \in \mathbb{R}^{|\mathcal{V}| \times D}$, where each row represents the distributed representation of dimension D for a word in a finite vocabulary $\mathcal{V}$. We construct $\mathcal{V}$ from the training data after filtering out the infrequent words.

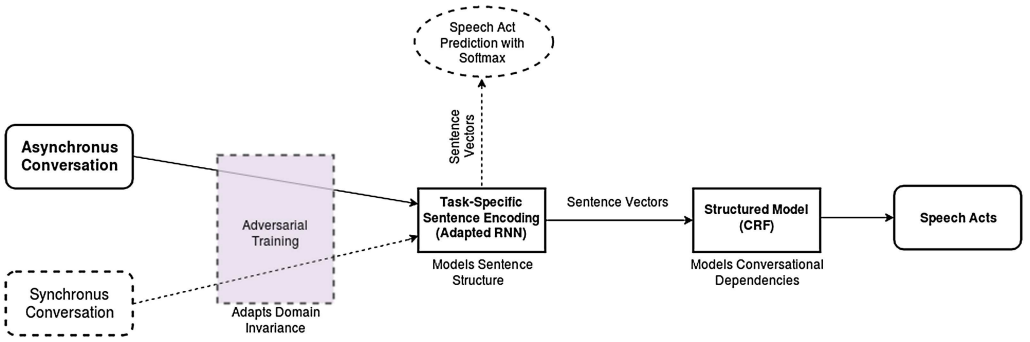
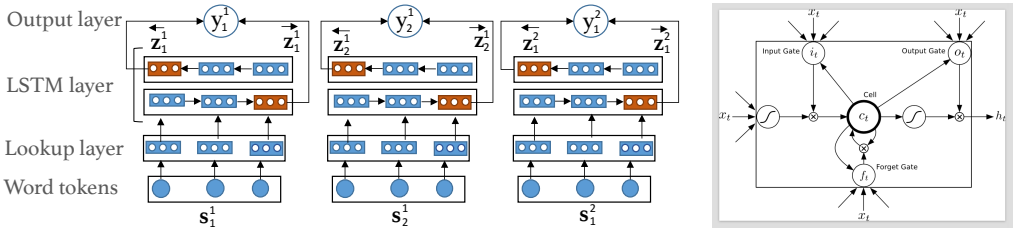


Figure 2
Our two-step inference framework for speech act recognition in asynchronous conversation. Each sentence in the conversation is first encoded into a task-specific representation by a recurrent neural network (RNN). The RNN is trained on the speech act classification task, and leverages large labeled data from synchronous domains (e.g., meetings) in an adversarial domain adaptation training method. A structured model (CRF) then takes the encoded sentence vectors as input, and performs joint prediction over all sentences in a conversation.



(a) Bidirectional LSTM-based RNN model

(b) An LSTM cell.

Figure 3
A bidirectional LSTM-RNN to encode each sentence $\mathbf{s}_m^n$ into a condensed vector $\mathbf{z}_\mathbf{m}^\mathbf{n}$. The network is trained to classify each sentence into its speech act type.

Given an input sentence $\mathbf{s} = (w_1, \cdots, w_T)$ of length T, we first map each word w_t to its corresponding index in E (equivalently, in $\mathcal{V}$). The first layer of our network is a look-up layer that transforms each of these indices to a distributed representation $\mathbf{x}_t \in \mathbb{R}^D$ by looking up the embedding matrix E. We consider E a model parameter to be learned by backpropagation. We can initialize E randomly or using pretrained word vectors (to be described in Section 4.2). The output of the look-up layer is a matrix in $\mathbb{R}^{T \times D}$, which is fed to the recurrent layer.

The recurrent layer computes a compositional representation $\overrightarrow{\mathbf{h}}_t$ at every time step t by performing nonlinear transformations of the current input $\mathbf{x}_t$ and the output of the previous time step $\overrightarrow{\mathbf{h}}_{t-1}$. We use LSTM blocks (Hochreiter and Schmidhuber 1997) in the recurrent layer. As shown in Figure 3(b), each LSTM block is composed of four elements: (i) a memory cell c (a neuron) with a self-connection, (ii) an input gate i to control the flow of input signal into the neuron, (iii) an output gate o to control the effect of neuron activation on other neurons, and (iv) a forget gate f to allow the neuron to adaptively reset its current state through a self-connection. The following sequence of equations describe how the memory blocks are updated at every time step t:

$$\mathbf{i}_t = \text{sigh}(U_i \mathbf{h}_{t-1} + V_i \mathbf{x}_t) \tag{1}$$

$$\mathbf{f}_t = \text{sigh}(U_f \mathbf{h}_{t-1} + V_f \mathbf{x}_t) \tag{2}$$

$$\mathbf{c}_t = \mathbf{i}_t \odot \tanh(U_c \mathbf{h}_{t-1} + V_c \mathbf{x}_t) + \mathbf{f}_t \odot \mathbf{c}_{t-1} \tag{3}$$

$$\mathbf{o}_t = \text{sigh}(U_o \mathbf{h}_{t-1} + V_o \mathbf{x}_t) \tag{4}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{5}$$

where U and V are the weight matrices between two consecutive hidden layers, and between the input and the hidden layers, respectively.[3] The symbols sigh and tanh denote hard sigmoid and hard tan nonlinear functions, respectively, and the symbol $\odot$ denotes an element-wise product of two vectors. LSTM-RNNs, by means of their specifically designed gates (as opposed to simple RNNs), are capable of capturing long-range dependencies. We can interpret $\mathbf{h}_t$ as an intermediate representation summarizing

3 There is bias associated with each nonlinear transformation, which we have omitted for notational simplicity.

the past, that is, the sequence $(w_1, w_2, \ldots, w_t)$. The output of the last time step $\mathbf{h}_T = \mathbf{z}$ can thus be considered as the representation of the entire sentence, which can be fed to the classification layer.

The classification layer uses a `softmax` for multi-class classification. Formally, the probability of the k-th class for classifying into K speech act classes is

$$p(y = k|\mathbf{s}, W, \theta) = \frac{\exp\ (\mathbf{w}_k^T \mathbf{z})}{\sum_{k=1}^{K} \exp\ (\mathbf{w}_k^T \mathbf{z})} \tag{6}$$

where W are the classifier weights, and $\theta = \{E, U, V\}$ are the encoder parameters. We minimize the negative log likelihood of the gold labels. The negative log likelihood for one data point is:

$$\mathcal{L}_c(W, \theta) = -\sum_{k=1}^{K} \mathcal{I}(y = k) \log p(y = k|\mathbf{s}, W, \theta) \tag{7}$$

where $\mathcal{I}(y = k)$ is an indicator function to encode the gold labels: $\mathcal{I}(y = k) = 1$ if the gold label $y = k$, otherwise 0.[4] The loss function minimizes the cross-entropy between the predicted distribution and the target distribution (i.e., gold labels).

Bidirectionality. The RNN just described encodes information that it obtains only from the past. However, information from the future could also be crucial for recognizing speech acts. This is especially true for longer sentences, where a unidirectional LSTM can be limited in encoding the necessary information into a single vector. Bidirectional RNNs (Schuster and Paliwal 1997) capture dependencies from both directions, thus providing two different views of the same sentence. This amounts to having a backward counterpart for each of the equations from (1) to (5). For classification, we use the concatenated vector $\mathbf{z} = [\overrightarrow{\mathbf{z}}, \overleftarrow{\mathbf{z}}]$ (equivalently, $[\overrightarrow{\mathbf{h}_T}, \overleftarrow{\mathbf{h}_T}]$), where $\overrightarrow{\mathbf{z}}$ and $\overleftarrow{\mathbf{z}}$ are the encoded vectors summarizing the past and the future, respectively.

3.2 Adapting LSTM-RNN with Adversarial Training

The LSTM-RNN described in the previous section can model long-distance dependencies between words, and, given enough training data, it should be able to compose a sentence, capturing its syntactic and semantic properties. However, when it comes to speech act recognition in asynchronous conversations, as mentioned before, not many large corpora annotated with a standard tagset are available. Because of the large number of parameters, the LSTM-RNN model usually overfits when it is trained on small data sets of asynchronous conversations (shown later in Section 5).

One solution to address this problem is to use data from synchronous domains for which large annotated corpora are available (e.g., MRDA meeting corpus). However, as we will see, although simple concatenation of data sets generally improves the performance of the LSTM-RNN model, it does not provide the optimal solution because the conversations in synchronous and asynchronous domains are different in modality (spoken vs. written) and in style. In other words, to get the best out of the available synchronous domain data, we need to adapt our model.

4 This is also known as one-hot vector representation.

Our goal is to adapt the LSTM-RNN encoder so that it learns to encode sentence representations $\mathbf{z}$ (i.e., features used for classification) that are not only discriminative for the act classification task, but also invariant across the domains. To this end, we propose to use the **domain adversarial training** of neural networks proposed recently by Ganin et al. (2016).

Let $\mathcal{D}_S = \{\mathbf{s}_n, y_n\}_{n=1}^{N}$ denote the set of N training instances (labeled) in the source domain (e.g., MRDA meeting corpus). We consider two possible adaptation scenarios:

(i) **Unsupervised adaptation:** In this scenario, we have only *unlabeled* examples in the target domain (e.g., forum). Let $\mathcal{D}_T^u = \{\mathbf{s}_n\}_{n=N+1}^{M}$ be the set of $(M - N - 1)$ unlabeled training instances in the target domain with M being the total number of training instances in the two domains.

(ii) **Supervised adaptation:** In addition to the unlabeled instances $\mathcal{D}_T^u$, here we have access to some *labeled* training instances in the target domain, $\mathcal{D}_T^l = \{\mathbf{s}_n, y_n\}_{n=M+1}^{L}$, with L being the total number of training examples in the two domains.

In the following, we describe our models for these two adaptation scenarios in turn.

3.2.1 Unsupervised Adaptation. Figure 4 shows our extended LSTM-RNN network trained for domain adaptation. The input sentence $\mathbf{s}$ is sampled either from a synchronous domain (e.g., meeting) or from an asynchronous (e.g., forum) domain. As before, we pass the sentence through a look-up layer and a bidirectional recurrent layer to encode it into a distributed representation $\mathbf{z} = [\overrightarrow{\mathbf{z}}, \overleftarrow{\mathbf{z}}]$, using our bidirectional LSTM-RNN encoder. For domain adaptation, our goal is to adapt the encoder to generate $\mathbf{z}$, such that it is not only informative for the target classification task (i.e., speech act recognition) but also invariant across domains. Upon achieving this, we can use the adapted LSTM-RNN encoder to encode a target sentence, and use the source classifier (the softmax layer) to classify the sentence into its corresponding speech act type.

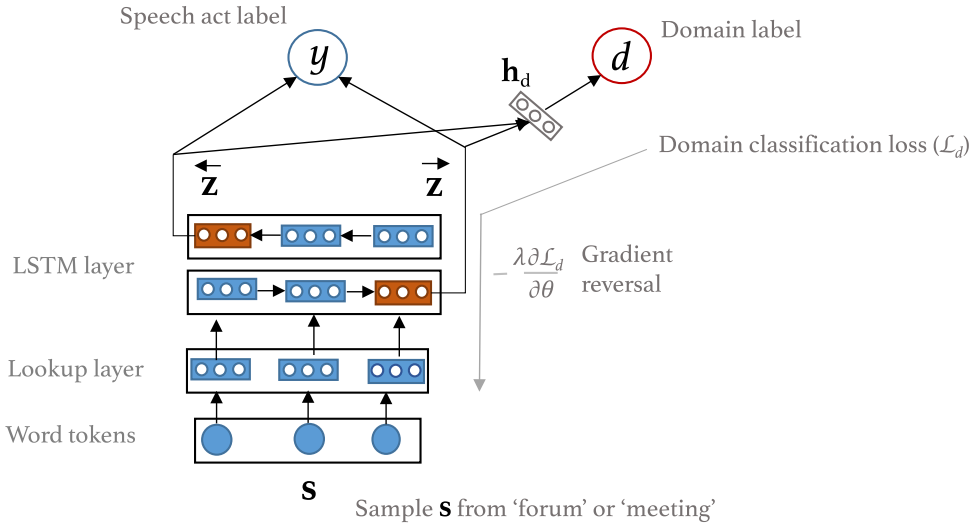


Figure 4
Adversarial LSTM-RNN for domain adaptation.

To this end, we add a domain discriminator, another neural network that takes $\mathbf{z}$ as input and tries to discriminate the domains of the input sentence (e.g., meeting vs. forum). Formally, the output of the domain discriminator is defined by a sigmoid function:

$$\hat{d}_\omega = p(d = 1|\mathbf{z}, \omega, \theta) = \text{sigm}(\mathbf{w}_d^T \mathbf{h}_d) \tag{8}$$

where $d \in \{0, 1\}$ denotes the domain of the sentence $\mathbf{s}$ (1 for meeting, 0 for forum), $\mathbf{w}_d$ are the final layer weights of the discriminator, and $\mathbf{h}_d = g(U_d \mathbf{z})$ defines the hidden layer of the discriminator with U_d being the layer weights, and $g(.)$ being the ReLU activations (Nair and Hinton 2010). We use the negative log-probability as the discrimination loss:

$$\mathcal{L}_d(\omega, \theta) = -d \log \hat{d}_\omega - (1 - d) \log \left(1 - \hat{d}_\omega\right) \tag{9}$$

The composite network (Figure 4) has three players: (i) the encoder (E), (ii) the classifier (C), and (iii) the discriminator (D). During training, the encoder and the classifier play a co-operative game, while the encoder and the discriminator play an adversarial game. The training objective of the composite model can be written as follows:

$$\mathcal{L}(W, \theta, \omega) = \underbrace{\sum_{n=1}^{N} \mathcal{L}_c^n(W, \theta)}_{\text{act classification (source)}} - \lambda \Big[\underbrace{\sum_{n=1}^{N} \mathcal{L}_d^n(\omega, \theta)}_{\text{domain discrimination (source)}} + \underbrace{\sum_{n=N+1}^{M} \mathcal{L}_d^n(\omega, \theta)}_{\text{domain discrimination (target)}} \Big] \tag{10}$$

where $\theta = \{E, U, V\}$ are the parameters of the LSTM-RNN encoder, W are the classifier weights, and $\omega = \{U_d, \mathbf{w}_d\}$ are the parameters of the discriminator network.[5] The hyper-parameter λ controls the relative strength of the two networks. In training, we look for parameter values that satisfy a *min-max* optimization criterion as follows:

$$\theta^* = \underset{W,\theta}{\text{argmin}} \max_{U_d, \mathbf{w}_d} \mathcal{L}(W, \theta, \omega) \tag{11}$$

which involves a maximization (gradient ascent) with respect to $\{U_d, \mathbf{w}_d\}$ and a minimization (gradient descent) with respect to θ and W. Maximizing $\mathcal{L}(W, \theta, \omega)$ with respect to $\{U_d, \mathbf{w}_d\}$ is equivalent to minimizing the discriminator loss $\mathcal{L}_d(\omega, \theta)$ in Equation (9), which aims to improve the discrimination accuracy. When put together, the updates of the shared encoder parameters $\theta = \{E, U, V\}$ for the two networks work *adversarially* with respect to each other.

In our gradient descent training, the min-max optimization is achieved by reversing the gradients (Ganin et al. 2016) of the domain discrimination loss $\mathcal{L}_d(\omega, \theta)$, when they are backpropagated to the encoder. As shown in Figure 4, the gradient reversal

5 For simplicity, we list U and V parameters of LSTM in a generic way rather than being specific to the gates.

is applied to the recurrent and embedding layers. This optimization set-up is related to the training method of Generative Adversarial Networks (Goodfellow et al. 2014), where the goal is to build deep *generative* models that can generate realistic images. The discriminator in Generative Adversarial Networks tries to distinguish real images from model-generated images, and thus the training attempts to minimize the discrepancy between the two image distributions. When backpropagating to the generator network, they consider a slight variation of the reverse gradients with respect to the discriminator loss. In particular, if $\hat{d}_\omega$ is the discriminator probability for real images, rather than reversing the gradients of $-\log(1-\hat{d}_\omega)$, they backpropagate the gradients of $-\log\hat{d}_\omega$ to the generator. Reversing the gradient is just a different way of achieving the same goal.

Algorithm 1 presents pseudocode of our training algorithm based on stochastic gradient descent (SGD). We first initialize the model parameters by sampling from Glorot-uniform distribution (Glorot and Bengio 2010). We then form minibatches of size b by randomly sampling $b/2$ labeled examples from $\mathcal{D}_S$ and $b/2$ unlabeled examples from $\mathcal{D}_T^u$. For labeled instances, both $\mathcal{L}_c(W,\theta)$ and $\mathcal{L}_d(\omega,\theta)$ losses are active, while only $\mathcal{L}_d(\omega,\theta)$ is active for unlabeled instances.

The main challenge in adversarial training is to balance the two components (the task classifier and the discriminator) of the network. If one component becomes smarter, its loss to the shared layer becomes useless, and the training fails to converge (Arjovsky, Chintala, and Bottou 2017). Equivalently, if one component gets weaker, its loss overwhelms that of the other, causing training to fail. In our experiments, we found the domain discriminator to be weaker; initially, it could not distinguish the domains often. To balance the two components, we would need the error signals from the discriminator to be fairly weak initially, with full power unleashed only as the classification errors start to dominate. We follow the weighting schedule proposed by Ganin et al. (2016, page 21), who initialize λ to 0, and then change it gradually to 1 as training progresses. That is, we start training the task classifier first, and we gradually add the discriminator's loss.

Algorithm 1: Model training with stochastic gradient descent.

Input : Data $\mathcal{D}_S = \{\mathbf{s}_n, y_n\}_{n=1}^N$, $\mathcal{D}_T^u = \{\mathbf{s}_n\}_{n=N+1}^M$ and batch size b
Output: Adapted model parameters $\theta = \{E, U, V\}$, W
1. Initialize model parameters;
2. **repeat**
   (a) Randomly sample $\frac{b}{2}$ labeled examples from $\mathcal{D}_S$
   (b) Randomly Sample $\frac{b}{2}$ unlabeled examples from $\mathcal{D}_T^u$
   (c) Compute $\mathcal{L}_c(W,\theta)$ and $\mathcal{L}_d(\omega,\theta)$
   (d) Set $\lambda = \frac{2}{1+\exp(-10*p)} - 1$; p is the training progress linearly changing form 0 to 1.
   `// Classifier & Encoder`
   (e) Take a gradient step for $\frac{2}{b}\nabla_{W,\theta}\mathcal{L}_c(W,\theta)$
   `// Discriminator`
   (f) Take a gradient step for $\frac{2\lambda}{b}\nabla_{U_d,\mathbf{w}_d}\mathcal{L}_d(\omega,\theta)$
   `// Gradient reversal to fool Discriminator`
   (g) Take a gradient step for $-\frac{2\lambda}{b}\nabla_\theta\mathcal{L}_d(\omega,\theta)$
until *convergence*;

3.2.2 Supervised Adaptation. It is quite straightforward to extend our adaptation method to a supervised setting, where we have access to some labeled instances in the target domain. Similar to the instances in the source domain ($\mathcal{D}_S$), the labeled instances in the target domain ($\mathcal{D}_T^l$) are used for act classification and domain discrimination. The total training loss in the supervised adaptation setting can be written as

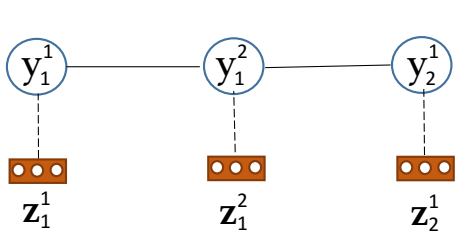
$$\mathcal{L}(W,\theta,\omega) = \underbrace{\sum_{n=1}^{N}\mathcal{L}_c^n(W,\theta)}_{\text{act classif. (source)}} + \underbrace{\sum_{n=M+1}^{L}\mathcal{L}_c^n(W,\theta)}_{\textit{act classif. (target)}} - \lambda\Big[\underbrace{\sum_{n=1}^{N}\mathcal{L}_d^n(\omega,\theta)}_{\text{dom classif. (source)}} + \underbrace{\sum_{n=N+1}^{L}\mathcal{L}_d^n(\omega,\theta)}_{\text{dom classif. (target)}}\Big] \tag{12}$$

where the second term is the classification loss on the labeled target data set $\mathcal{D}_T^l$, and the last term is the discrimination loss on both labeled and unlabeled data in the target domain. We modify the training algorithm accordingly. Specifically, each minibatch in SGD training is formed by labeled instances from both $\mathcal{D}_S$ and $\mathcal{D}_T^l$, and unlabeled instances from $\mathcal{D}_T^u$.

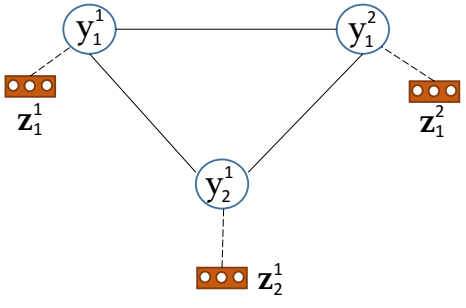
3.3 Conditional Structured Model for Conversational Dependencies

Given the vector representation of the sentences in an asynchronous conversation, we explore two different approaches to learn classification functions. The first and the traditional approach is to learn a local classifier, ignoring the structure in the output and using it for predicting the label of each sentence separately. Indeed, this is the approach we took in the previous subsection when we fed the output layer of the LSTM RNNs (Figures 3 and 4) with the sentence vectors. However, this approach does not model the conversational dependency between sentences in a conversation (e.g., *adjacency* relations between question-answer and request-accept pairs).

The second approach, which we adopt in this article, is to model the dependencies between the output variables (i.e., speech act labels of the sentences), while learning the classification functions jointly by optimizing a global performance criterion. We represent each conversation by a graph $G = (V, E)$, as shown in Figure 5. Each node $i \in V$ is associated with an input vector $\mathbf{z}_i = \mathbf{z}_m^n$ (extracted from the LSTM-RNN), representing



(a) A linear chain CRF



(b) A fully connected CRF

Figure 5
Examples of conditional structured models for speech act recognition in asynchronous conversation. The sentence vectors ($\mathbf{z}_m^n$) are extracted from the LSTM-RNN model.

the encoded features for the sentence $\mathbf{s}_m^n$, and an output variable $y_i \in \{1, 2, \cdots, K\}$, representing the speech act type. Similarly, each edge $(i, j) \in E$ is associated with an input feature vector $\phi(\mathbf{z}_i, \mathbf{z}_j)$, derived from the node-level features, and an output variable $y_{i,j} \in \{1, 2, \cdots, L\}$, representing the state transitions for the pair of nodes. We define the following conditional joint distribution:

$$p(\mathbf{y}|\mathbf{v}, \mathbf{w}, \mathbf{z}) = \frac{1}{Z(\mathbf{v}, \mathbf{w}, \mathbf{z})} \prod_{i \in V} \underbrace{\psi_n(y_i|\mathbf{z}, \mathbf{v})}_{\text{node factor}} \prod_{(i,j) \in E} \underbrace{\psi_e(y_{i,j}|\mathbf{z}, \mathbf{w})}_{\text{edge factor}} \tag{13}$$

where ψ_n and ψ_e are the node and the edge *factors*, and $Z(.)$ is the global normalization constant that ensures a valid probability distribution. We use a log-linear representation for the factors:

$$\psi_n(y_i|\mathbf{z}, \mathbf{v}) = \exp(\mathbf{v}^T \phi(y_i, \mathbf{z})) \tag{14}$$

$$\psi_e(y_{i,j}|\mathbf{z}, \mathbf{w}) = \exp(\mathbf{w}^T \phi(y_{i,j}, \mathbf{z})) \tag{15}$$

where $\phi(.)$ is a feature vector derived from the inputs and the labels. This model is essentially a pairwise conditional random field (Murphy 2012). The global normalization allows CRFs to surmount the so-called label bias problem (Lafferty, McCallum, and Pereira 2001), allowing them to take long-range interactions into account. The log likelihood for one data point $(\mathbf{z}, \mathbf{y})$ (i.e., a conversation) is:

$$f(\theta) = \sum_{i \in V} \mathbf{v}^T \phi(y_i, \mathbf{z}) + \sum_{(i,j) \in E} \mathbf{w}^T \phi(y_{i,j}, \mathbf{z}) - \log Z(\mathbf{v}, \mathbf{w}, \mathbf{z}) \tag{16}$$

This objective is convex, so we can use gradient-based methods to find the global optimum. The gradients have the following form:

$$f'(\mathbf{v}) = \sum_{i \in V} \phi(y_i, \mathbf{z}) - \mathbb{E}[\phi(y_i, \mathbf{z})] \tag{17}$$

$$f'(\mathbf{w}) = \sum_{(i,j) \in E} \phi(y_{i,j}, \mathbf{z}) - \mathbb{E}[\phi(y_{i,j}, \mathbf{z})] \tag{18}$$

where the $\mathbb{E}[\phi(.)]$ denote the expected feature vectors. In our case, the node or sentence features are the task-specific sentence embeddings extracted from the bi-directional LSTM-RNN model (possibly domain adapted by adversarial training), and for edge features, we use the hadamard product (i.e., element-wise product) of the two corresponding node vectors.

3.3.1 Training and Inference in CRFs. Traditionally, CRFs have been trained using offline methods like limited-memory BFGS (Murphy 2012). Online training of CRFs using SGD was proposed by Vishwanathan et al. (2006). Because RNNs are trained with online methods, to compare our two methods, we use an SGD-based algorithm to train our CRFs. Algorithm 2 gives the pseudocode of the training procedure.

We use Belief Propagation (BP) (Pearl 1988) for inference in our CRFs. BP is guaranteed to converge to an exact solution if the graph is a tree. However, exact inference is intractable for graphs with loops. Despite this, Pearl (1988) advocates for BP in loopy

Algorithm 2: Online learning algorithm for conditional random fields.

1. Initialize the model parameters $\mathbf{v}$ and $\mathbf{w}$;
2. **repeat**
   for *each thread* $G = (V, E)$ **do**
      (*a*) Compute node and edge factors $\psi_n(y_i|\mathbf{z}, \mathbf{v})$ and $\psi_e(y_{i,j}|\mathbf{z}, \mathbf{w})$;
      (*b*) Infer node and edge marginals using sum-product loopy BP;
      (*c*) Update: $\mathbf{v} = \mathbf{v} - \eta \frac{1}{|V|} f'(\mathbf{v})$;
      (*d*) Update: $\mathbf{w} = \mathbf{w} - \eta \frac{1}{|E|} f'(\mathbf{w})$;
   end
until *convergence*;

graphs as an approximation (see Murphy 2012, page 768). The algorithm is then called **loopy BP**. Although loopy BP gives approximate solutions for general graphs, it often works well in practice (Murphy, Weiss, and Jordan 1999), outperforming other methods such as mean field (Weiss 2001).

3.3.2 Variations of Graph Structures. One of the advantages of the pairwise CRF in Equation (13) is that we can define this model over arbitrary graph structures, which allows us to capture conversational dependencies at various levels. Modeling the arbitrary graph structure can be crucial, especially in scenarios where the reply-to structure of the conversation is not known. By defining structured models over plausible graph structures, we can get a sense of the underlying conversational structure. We distinguish between two types of conversational dependencies:

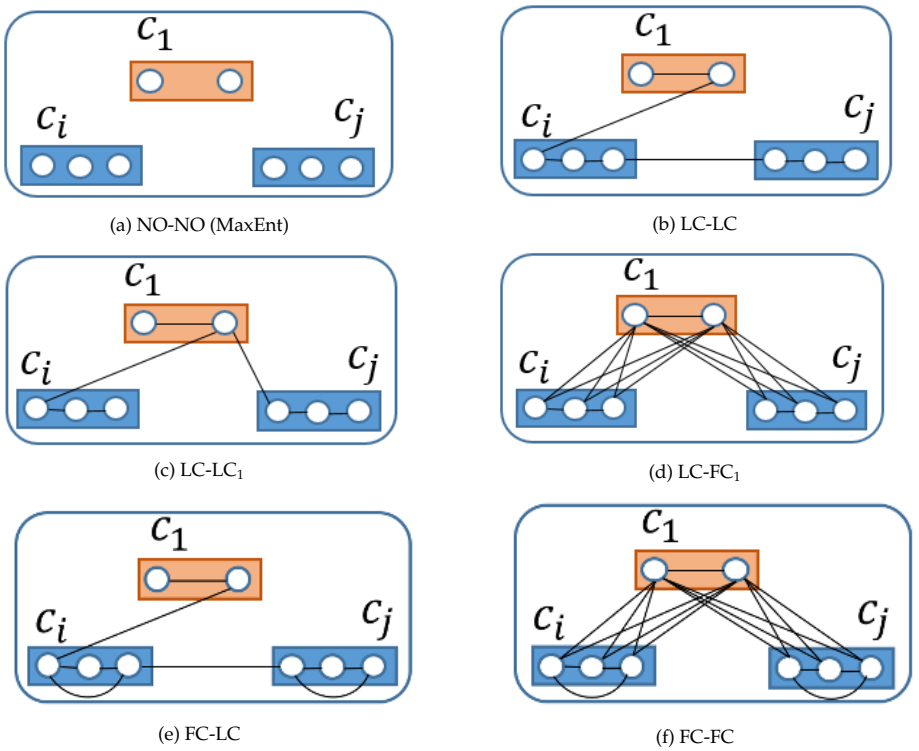
(i) **Intra-comment connections:** This defines how the speech acts of the sentences inside a comment are connected with each other.

(ii) **Across-comment connections:** This defines how the speech acts of the sentences across comments are connected in a conversation.

Table 1 summarizes the connection types that we have explored in our CRF models. Each configuration of intra- and across-connections yields a different pairwise CRF. Figure 6 shows four such CRFs with three comments — C_1 being the first comment, and C_i and C_j being two other comments in the conversation. Figure 6(a) shows the structure for the **NO-NO** configuration, where there is no link between nodes of both intra- and across-comments. In this setting, the CRF model boils down to the MaxEnt model. Figure 6(b) shows the structure for **LC-LC** configuration, where there are linear

Table 1
Connection types in CRF models.

Tag	Connection type	Applicable to
NO	No connection between nodes	intra & across
LC	Linear chain connection	intra & across
FC	Fully connected	intra & across
FC_1	Fully connected with first comment only	across
LC_1	Linear chain with first comment only	across



(a) NO-NO (MaxEnt) (b) LC-LC

(c) LC-LC$_1$ (d) LC-FC$_1$

(e) FC-LC (f) FC-FC

Figure 6
CRFs over different graph structures.

chain relations between nodes of both intra- and across-comments. The linear chain across comments refers to the structure, where the last sentence of each comment is connected to the first sentence of the comment that comes next in the temporal order.

Figures 6(c) shows the CRF for **LC-LC$_1$**, in which the sentences inside a comment have linear chain connections, and the last sentence of the first comment is connected to the first sentence of the other comments. Figure 6(d) shows the graph structure for **LC-FC$_1$** configuration, in which the sentences inside comments have linear chain connections, and sentences of the first comment are fully connected with the sentences of the other comments. Similarly, Figures 6(e) and 6(f) show the graph structures for **FC-LC** and **FC-FC** configurations.

4. Corpora

In this section, we describe the data sets used in our experiments. We use a number of *labeled* data sets to train and test our models, one of which we constructed in this work. Additionally, we use a large *unlabeled* conversational data set to train our (unsupervised) word embedding models.

4.1 Labeled Corpora

There exist large corpora of utterances annotated with speech acts in synchronous spoken domains, for example, Switchboard-DAMSL (SWBD) (Jurafsky, Shriberg, and

Table 2
Dialog act tags and their relative frequencies in the BC3 and TripAdvisor (TA) corpora.

Tag	Description	BC3	TA
S	Statement	69.56%	65.62%
P	Polite mechanism	6.97%	9.11%
QY	Yes-no question	6.75%	8.33%
AM	Action motivator	6.09%	7.71%
QW	Wh-question	2.29%	4.23%
A	Accept response	2.07%	1.10%
QO	Open-ended question	1.32%	0.92%
AA	Acknowledge and appreciate	1.24%	0.46%
QR	Or/or-clause question	1.10%	1.16%
R	Reject response	1.06%	0.64%
U	Uncertain response	0.79%	0.65%
QH	Rhetorical question	0.75%	0.08%

Biasca 1997) and Meeting Recorder Dialog Act (MRDA) (Dhillon et al. 2004). However, the asynchronous domain lacks such large corpora. Some prior studies (Cohen, Carvalho, and Mitchell 2004; Feng et al. 2006; Ravi and Kim 2007; Bhatia, Biyani, and Mitra 2014) tackle the task at the comment level, and use task-specific tagsets. In contrast, in this work we are interested in identifying speech acts at the sentence level, and also using a standard tagset like the ones defined in SWBD or MRDA.

Several studies attempt to solve the task at the sentence level. Jeong, Lin, and Lee (2009) created a data set of TripAdvisor (TA) forum conversations annotated with the standard 12 act types defined in MRDA. They also remapped the BC3 e-mail corpus (Ulrich, Murray, and Carenini 2008) according to this tagset. Table 2 shows the tags and their relative frequency in the two data sets. Subsequent studies (Joty, Carenini, and Lin 2011; Tavafi et al. 2013; Oya and Carenini 2014) use these data sets. We also use these data sets in our work. Table 3 shows some basic statistics about these data sets. On average, BC3 conversations are longer than those of TripAdvisor in terms of both number of comments and number of sentences.

Since these data sets are relatively small in size with sparse tag distributions, we group the 12 act types into 5 coarser classes to learn a reasonable classifier. Some prior work (Tavafi et al. 2013; Oya and Carenini 2014) has also taken the same approach. More specifically, all the question types are grouped into one general class *Question*, all response types into *Response*, and appreciation and polite mechanisms into the *Polite* class.

In addition to the asynchronous data sets – TA, BC3, and QC3 (to be introduced subsequently), we also demonstrate the performance of our models on the synchronous

Table 3
Statistics about TripAdvisor (TA), BC3, and QC3 corpora.

	Asynchronous		
	TA	BC3	QC3
Total number of conversations	200	39	47
Average number of comments per conversation	4.02	6.54	13.32
Average number of sentences per conversation	18.56	34.15	33.28
Average number of words per sentence	14.90	12.61	19.78

Table 4
Distribution of speech acts (in percentage) in our corpora.

		Asynchronous			Synchronous
Tag	Description	TA	BC3	QC3	MRDA
SU	Suggestion (Action motivator)	7.71	5.48	17.38	5.97
R	Response (Accept, Reject, Uncertain)	2.4	3.75	5.24	15.63
Q	Questions (Yes-no, Wh, Rhetorical, Or-clause, Open-ended)	14.71	8.41	12.59	8.62
P	Polite (Acknowledge & appreciate, Polite)	9.57	8.63	6.13	3.77
ST	Statement	65.62	73.72	58.66	66.00

Table 5
Cohen's κ agreement for different speech acts in QC3.

Tag	Speech act	Cohen's κ
SU	Suggestion	0.86
R	Response	0.43
Q	Question	0.87
P	Polite	0.75
ST	Statement	0.78

MRDA meeting corpus, and use it for domain adaptation. Table 4 shows the label distribution of the resulting data sets; Statement is the most dominant class, followed by Question, Polite, and Suggestion.

4.1.1 QC3 Conversational Corpus: A New Asynchronous Data Set. Because both TripAdvisor and BC3 are quite small to make a general comment about model performance in asynchronous conversations, we have created a new annotated data set of forum conversations called Qatar Computing Conversational Corpus or QC3.[6] We selected 50 conversations from a popular community question answering site named Qatar Living[7] for our annotation. We used three conversations for our pilot study and used the remaining 47 for the actual study. The resulting corpus, as shown in the last column of Table 3, on average contains 13.32 comments and 33.28 sentences per conversation, and 19.78 words per sentence.

Two native speakers of English annotated each conversation using a Web-based annotation framework (Ulrich, Murray, and Carenini 2008). They were asked to annotate each sentence with the most appropriate speech act tag from the list of five speech act types. Because this task is not always obvious, we gave them detailed annotation guidelines with real examples. We use Cohen's κ to measure the agreement between the annotators. The third column in Table 5 presents the κ values for the act types, which vary from 0.43 (for Response) to 0.87 (for Question).

In order to create a consolidated data set, we collected the disagreements between the two annotators, and used a third annotator to resolve those cases. The fifth column

6 Available from https://ntunlpsg.github.io/project/speech-act/.
7 http://www.qatarliving.com/.

Table 6
Data sets and their statistics used for training the conversational word embeddings.

	Domain	Data sets	Number of Threads	Number of Tokens	Number of Words
Asynchronous	E-mail	W3C	23,940	21,465,830	546,921
	Forum	TripAdvisor	25,000	2,037,239	127,233
	Forum	Qatar Living	219,690	103,255,922	1,157,757
Synchronus	Meeting	MRDA	-	675,110	18,514
	Phone	SWBD	-	1,131,516	57,075

in Table 4 presents the distribution of the speech acts in the resulting data set. As we can see, after Statement, Suggestion is the most frequent class, followed by the Question and the Polite classes.

4.2 Conversational Word Embeddings

One simple way to exploit unlabeled data for semi-supervised learning is to use word embeddings that are learned from large unlabeled data sets (Turian, Ratinov, and Bengio 2010). Word embeddings such as word2vec skip-gram (Mikolov, Yih, and Zweig 2013) and Glove vectors (Pennington, Socher, and Manning 2014) capture syntactic and semantic properties of words and their linguistic regularities in the vector space. The skip-gram model was trained on part of the Google news data set containing about 100 billion words, and it contains 300-dimensional vectors for 3 million unique words and phrases.[8] Glove was trained on the combination of Wikipedia 2014 and Gigaword 5 data sets containing 6B tokens and 400K unique (uncased) words. It comes with 50d, 100d, 200d, and 300d vectors.[9] For our experiments, we use the 300d vectors.

Many recent studies have shown that the pretrained embeddings improve the performance on supervised tasks (Schnabel et al. 2015). In our work, we have used these generic off-the-shelf pretrained embeddings to boost the performance of our models. In addition, we have also trained the word2vec skip-gram model and Glove on a large conversational corpus to obtain more relevant *conversational* word embeddings. Later in our experiments (Section 5) we will demonstrate that the *conversational* word embeddings are more effective than the generic ones because they are trained on similar data sets.

To train the word embeddings, we collected conversations of both synchronous and asynchronous types. For asynchronous, we collected e-mail threads from W3C (w3c.org), and forum conversations from TripAdvisor and QatarLiving sites. The raw data was too noisy to directly inform our models, as it contains system messages and signatures. We cleaned up the data with the intention of keeping only the headers, bodies, and quotations. For synchronous, we used the utterances from the SWBD and MRDA corpora. Table 6 shows some basic statistics about these (unlabeled) data sets. We trained our word vectors on the concatenated set of all data sets (i.e., 120M tokens). Note that the conversations in our labeled data sets were taken from these sources (e.g., BC3 from W3C, QC3 from QatarLiving, and TA from TripAdvisor.)

8 Available from `https://code.google.com/archive/p/word2vec/`.
9 Available from `https://nlp.stanford.edu/projects/glove/`.

Table 7
Roadmap to our experiments.

Model Tested	Training Regime	Section	Corpora Used
LSTM-RNN	In-domain supervised	5.2.2	QC3/TA/BC3/MRDA (all labeled)
	Concatenation supervised	5.2.3	QC3+TA+BC3+MRDA (labeled)
	Unsup. adaptation	5.2.4	QC3/TA/BC3 (unlabeled) + MRDA (labeled)
	Semi-sup. adaptation	5.2.4	QC3/TA/BC3 (labeled) + MRDA (labeled)
CRFs	In-domain supervised	5.3	QC3/TA/BC3 (labeled; conversation level)

5. Experiments

In this section, we present our experimental settings, results, and analysis. We start with an outline of the experiments.

5.1 Outline of Experiments

Our main objective is to evaluate our speech act recognizer on asynchronous conversations. For this, we evaluate our models on the forum and e-mail data sets introduced earlier in Section 4.1: (i) our newly created QC3 data set, (ii) the TripAdvisor (TA) data set from Jeong, Lin, and Lee (2009), and (iii) the BC3 e-mail corpus from (Ulrich, Murray, and Carenini 2008). In addition, we validate our sentence encoding approach on the MRDA meeting corpus.

Because of the noisy and informal nature of conversational texts, we performed a series of preprocessing steps before using it for training or testing. We normalize all characters to their lowercased forms, truncate elongations to two characters, and spell out every digit and URL. We further tokenized the texts using the CMU TweetNLP tool (Gimpel et al. 2011).

For performance comparison, we use both **accuracy** and **macro-averaged $\mathbf{F_1}$** score. Accuracy gives the overall performance of a classifier but could be biased toward the most populated classes, whereas macro-averaged F_1 weights every class equally, and is not influenced by class imbalance. Statistical significance tests are done using an **approximate randomization** test based on the accuracy.[10] We used SIGF V.2 (Padó 2006) with 10,000 iterations.

In the following, we first demonstrate the effectiveness of our LSTM-RNN model for learning task-specific sentence encoding by training it on the task in three different settings: (i) training on **in-domain** data only, (ii) training on a **simple concatenation** of synchronous and asynchronous data, and (iii) training it with **adversarial training** for domain adaptation. We also compare the effectiveness of different embedding types in these three training settings. The best task-specific embeddings are then extracted and fed into the CRF models to learn inter-sentence dependencies. In Section 5.3, we compare how our CRF models with different conversational graph structure perform. Table 7 gives an outline of our experimental roadmap.

10 Significance tests operate on individual instances rather than individual classes; thus not applicable for macro F_1.

Table 8
Number of sentences in train, development, and test sets for different data sets.

Corpora	Type	Train	Dev.	Test
QC3	asynchronous	1,252	157	156
TA	asynchronous	2,968	372	371
BC3	asynchronous	1,065	34	133
MRDA	synchronous	50,865	8,366	10,492
Total (CONCAT)	asynchronous + synchronous	56,150	8,929	11,152

5.2 Effectiveness of LSTM RNN

We first describe the experimental settings for our LSTM RNN sentence encoding model—the data set splits, training settings, and compared baselines. Then we present our results on the three training scenarios as outlined in Table 7.

5.2.1 Experimental Settings. We split each of our asynchronous corpora randomly into 70% *sentences* for training, 10% for development, and 20% for testing. For MRDA, we use the same train:test:dev split as Jeong, Lin, and Lee (2009). Table 8 summarizes the resulting data sets.

We compare the performance of our LSTM-RNN model with MaxEnt (**ME**) and Multi-layer Perceptron (**MLP**) with one hidden layer.[11] In one setting, we fed them with the bag-of-words (**BOW**) representation of the sentence, namely, vectors containing binary values indicating the presence or absence of a word in the training set vocabulary. In another setting, we use a concatenation of the pretrained word embeddings as the sentence representation.

We train the models by optimizing the cross entropy in Equation (7) using the gradient-based learning algorithm ADAM (Kingma and Ba 2014).[12] The learning rate and other parameters were set to the values as suggested by the authors. To avoid overfitting, we use dropout (Srivastava et al. 2014) of hidden units and early-stopping based on the loss on the development set.[13] Maximum number of epochs was set to 50 for RNNs, ME, and MLP. We experimented with dropout rates of $\{0.0, 0.2, 0.4\}$, minibatch sizes of $\{16, 32, 64\}$, and hidden layer units of $\{100, 150, 200\}$ in MLP and LSTMs. The vocabulary $\mathcal{V}$ in LSTMs was limited to the most frequent P% ($P \in \{85, 90, 95\}$) words in the training corpus, where P is considered a hyperparameter.

We initialize the word vectors in our model either by sampling randomly from the small uniform distribution $\mathcal{U}(-0.05, 0.05)$, or by using pretrained embeddings. The dimension for random initialization was set to 128. For pretrained embeddings, we experiment with off-the-shelf embeddings that come with word2vec (Mikolov et al. 2013b) and Glove (Pennington, Socher, and Manning 2014) as well as with our conversational word embeddings (Section 4.2).

We experimented with four variations of our LSTM-RNN model: (i) **U-LSTM$_{\text{rand}}$**, referring to *unidirectional* RNN with *random* word vector initialization; (ii) **U-LSTM$_{\text{pre}}$**,

11 More hidden layers worsened the performance.
12 Other algorithms like Adagrad or RMSProp gave similar results.
13 l_1 and l_2 regularization on weights did not work well.

referring to *unidirectional* RNN initialized with pretrained word embeddings of type *pre*; (iii) **B-LSTM$_{\text{rand}}$**, referring to *bidirectional* RNN with *random* initialization; and (iv) **B-LSTM$_{\text{pre}}$**, referring to *bidirectional* RNN initialized with pretrained word vectors of type *pre*.

5.2.2 Results for In-Domain Training. Before reporting the performance of our sentence encoding model on asynchronous domains, we first evaluate it on the (synchronous) MRDA meeting corpus where it can be compared to previous studies on a large data set.

Results on MRDA Meeting Corpus. Table 9 presents the results on MRDA for in-domain training. The first two rows show the best results reported so far on this data set from Jeong, Lin, and Lee (2009) for classifying sentences into 12 speech act types; the first row shows the results of the model that uses only *n*-grams, and the second row shows the results using all of the features, including *n*-grams, speaker, part-of-speech, and dependency structure. Note that our LSTM RNNs and their *n*-gram model use the same word sequence information.

The second group of results (third and fourth rows) are for ME and MLP models with BOW sentence representation. The third group shows the results for unidirectional LSTM with random and pretrained off-the-shelf embeddings. The fourth group shows the corresponding results for bi-directional LSTMs. Finally, the fifth row presents the results for bi-directional LSTM with our conversational embeddings. To compare our results with the results of Jeong, Lin, and Lee (2009), we ran our models on 12-class classification task in addition to our original 5-class task.

It can be observed that all of our LSTM-RNNs achieve state-of-the-art results, and the bi-directional ones with pretrained embeddings generally perform better than others in terms of the F_1-score. The best results are obtained with our conversational embeddings. Our best model B-LSTM$_{\text{conv-glove}}$ (B-LSTM with Glove conversational embeddings) gives absolute improvements of about 5.0% and 3.5% in F_1 compared to the *n*-gram and all-features models, respectively, of Jeong, Lin, and Lee (2009). This is

Table 9
Results on MRDA (synchronous) meeting corpus in macro-averaged F_1 and accuracy. Accuracy numbers are shown in parentheses. Top two rows report results from Jeong, Lin, and Lee (2009) for their model with *n*-gram and all feature sets. Best results are **boldfaced**. Accuracy numbers significantly superior to the best baselines are marked with *.

		MRDA	
	Pretrained Embedding	5 classes	12 classes
Jeong, Lin, and Lee (2009) (n-gram)	-	-	57.53 (83.30)
Jeong, Lin, and Lee (2009) (all features)	-	-	59.04 (83.49)
ME$_{\text{bow}}$	-	65.25 (83.95)	57.79 (82.84)
MLP$_{\text{bow}}$	-	68.12 (84.24)	58.19 (83.24)
U-LSTM$_{\text{random}}$	-	71.19 (84.38)	58.72 (83.34)
U-LSTM$_{\text{google-w2v}}$	word2vec (Google)	72.32 (84.19)	59.05 (83.26)
U-LSTM$_{\text{glove}}$	Glove (off-the-shelf)	72.24 (84.93)	60.02 (83.14)
B-LSTM$_{\text{random}}$	-	71.26 (84.12)	60.98 (83.04)
B-LSTM$_{\text{google-w2v}}$	word2vec (Google)	72.34 (84.39)	61.72 (83.17)
B-LSTM$_{\text{glove}}$	Glove (off-the-shelf)	72.41 (84.80)	62.33 (82.82)
B-LSTM$_{\text{conv-w2v}}$	word2vec (conversation)	72.13 (85.42*)	62.18 (83.23)
B-LSTM$_{\text{conv-glove}}$	Glove (conversation)	**72.88 (85.43*)**	**62.53** (**83.61**)

Table 10
Results for in-domain training on QC3, TA, and BC3 asynchronous data sets in macro-averaged F_1 and accuracy (in parentheses). Best results are **boldfaced**. Accuracy numbers significantly superior to the best baselines are marked with *.

	QC3		TA		BC3	
	Testset	5 folds	Testset	5 folds	Testset	5 folds
ME_{bow}	55.11 (76.28)	55.15 (73.16)	62.82 (82.47)	62.65 (85.04)	54.37 (84.47)	52.69 (81.78)
MLP_{bow}	56.71 (74.35)	59.72 (72.46)	70.45 (83.83)	65.18 (84.02)	**63.98 (84.58)**	**62.37 (82.04)**
U-$LSTM_{random}$	54.52 (70.51)	53.39 (67.22)	64.52 (80.32)	59.20 (80.06)	44.41 (81.95)	42.21 (72.44)
U-$LSTM_{glove}$	59.95 (72.44)	55.56 (70.03)	67.70 (83.83)	60.82 (83.22)	45.67 (78.95)	43.75 (73.50)
U-$LSTM_{conv\text{-}glove}$	60.59 (75.64)	58.70 (72.78)	69.48 (83.56)	64.64 (83.39)	53.51 (84.21)	49.67 (77.71)
B-$LSTM_{random}$	57.57 (74.35)	58.24 (72.46)	74.70 (86.25*)	67.08 (84.53)	47.12 (81.20)	44.97 (77.59)
B-$LSTM_{glove}$	59.16 (73.07)	58.86 (72.45)	75.49 (86.77*)	68.31 (83.81)	51.15 (84.21)	50.67 (75.59)
B-$LSTM_{conv\text{-}glove}$	**64.72 (77.56*)**	**63.47 (75.59*)**	**76.15 (86.52*)**	**69.59 (86.18*)**	61.44 (83.45)	55.84 (79.95)

remarkable because our LSTM-RNNs learn the sentence representation automatically from the word sequence and do not use any hand-engineered features.

Results on Asynchronous Data Sets. Now let us consider the results in Table 10 for the asynchronous data sets—QC3, TA, and BC3. We show the results of our models based on 5-fold cross validation in addition to the random (20%) test set in Table 8. The 5-fold setting allows us to get more generic performance of the models on a particular data set. For simplicity, we only report the results for Glove embeddings that were found to be superior to word2vec embeddings.

We can observe trends similar to those for MRDA: (i) bidirectional LSTMs outperform their unidirectional counterparts, (ii) pretrained Glove vectors provide better results than the randomly initialized ones, and (iii) conversational word embeddings give the best results among the embedding types. When we compare these results with those of the baselines (ME_{bow} and MLP_{bow}), we see our method outperforms those on QC3 and TA (3.8% to 8.0%), but fails to do so on BC3. This is due to the small size of the data that affects deep neural methods like LSTM-RNNs, which usually require much labeled data to learn an effective compositional model. In the following, we show the effect of adding more labeled data from the MRDA meeting corpus.

5.2.3 Adding Meeting Data. To validate our claim that LSTM-RNNs can learn a more effective model for our task when they are provided with enough training data, we create a concatenated *training* setting by merging the training and the development sets of the four corpora in Table 8 (see the Train and Dev. columns in the last row); the test set for each data set remains the same. We will refer to this train-test setting as CONCAT.

Table 11 shows the results of the baseline and the B-LSTM models on the three test sets for this concatenated training setting. We notice that our B-LSTM models with pretrained embeddings outperform ME_{bow} and MLP_{bow} significantly. Again, the conversational Glove embeddings prove to be the best word vectors giving the best results across the data sets. Our best model gives absolute improvements of 2% to 12% in F_1 across the data sets over the best baselines.

When we compare these results with those in Table 10, we notice that with more heterogeneous data sets, B-LSTM, by virtue of its distributed and condensed representation, generalizes well across different domains. In contrast, ME and MLP, because of their BOW representation, suffer from the data diversity of different domains. These

Table 11
Macro-averaged F_1 and Accuracy (in parentheses) results for training on the **concatenated (CONCAT)** data set **without any explicit domain adaptation**. Best results are **boldfaced**. Accuracy numbers significantly higher than the best baseline MLP_{bow} are marked with *.

	Pretrained Emb	QC3 (Testset)	TA (Testset)	BC3 (Testset)
ME_{bow}	-	50.64 (71.15)	72.49 (84.10)	53.17 (76.00)
MLP_{bow}	-	58.60 (74.36)	73.07 (85.29)	56.19 (78.00)
B-$LSTM_{google\text{-}w2v}$	word2vec (off-the-shelf)	67.00 (79.49*)	74.63 (87.67*)	56.55 **(80.04*)**
B-$LSTM_{glove}$	Glove (off-the-shelf)	62.71 (80.13*)	76.61 (87.33*)	54.87 (80.00*)
B-$LSTM_{conv\text{-}w2v}$	word2vec (conversation)	66.34 (79.48*)	75.03 (86.55*)	**58.28** (79.00*)
B-$LSTM_{conv\text{-}glove}$	Glove (conversation)	**70.51 (80.77*)**	**78.08 (88.95*)**	57.47 (80.00*)

results also confirm that B-LSTM gives better sentence representation than BOW when it is given enough training data.

Comparison with Other Classifiers and Sentence Encoders. Now, we compare our best B-LSTM model (i.e., B-$LSTM_{conv\text{-}glove}$) with other classifiers and sentence encoders in the concatenated (CONCAT) training setting. The models that we compare with are:

(a) **$ME_{conv\text{-}glove}$:** We represent each sentence as a *concatenated* vector of its word vectors, and train a MaxEnt (ME) classifier based on this representation. For word vectors, we use our best performing conversational Glove vectors as we use in our B-$LSTM_{conv\text{-}glove}$ model. We set a maximum sentence length of 100 words, and used *zero-padding* for shorter sentences. This model has a total of 100 (input words) $\times$ 300 (embedding dimensions) $\times$ 5 (class labels) = 150,000 trainable parameters.[14]

(b) **$MLP_{conv\text{-}glove}$:** We represent each sentence similarly as above, and train a one-hidden layer Multi-layer Perceptron (MLP) based on the representation. The hidden layer has $1,000$ units, which is determined based on the performance on the development set. This model has a total of $100 \times 300 \times 1000 \times 5 = 150{,}000{,}000$ parameters.

(c) **$ME_{conv\text{-}glove\text{-}averaging}$:** We represent each sentence as a *mean* vector of its word vectors, and train a MaxEnt classifier using this representation. This model has a total of $300 \times 5 = 1{,}500$ trainable parameters.

(d) **$SVM_{conv\text{-}glove\text{-}averaging}$:** We train a SVM classifier based on the mean vector.[15] In our training, we use a linear kernel with the default C value of 1.0.

(e) **$ME_{skip\text{-}thought}$:** We encode each sentence with the skip-thought encoder of Kiros et al. (2015). The skip-thought model uses an encoder-decoder framework to learn the sentence representation in a *task-agnostic* (unsupervised) way. It encodes each sentence with a GRU-RNN (Cho et al. 2014), and uses the encoded vector to decode the words of the neighboring sentences using another GRU-based RNN as a language model. The model is originally trained on the BookCorpus[16] with a vocabulary size of 20K words. It then uses the CBOW word2vec vectors (Mikolov et al. 2013a) to expand the vocabulary size to 930,911 words. Following the recommendation from

14 For simplicity, we excluded the bias vectors from our computation.
15 SVM training with linear kernel did not scale to the concatenated vector.
16 `http://yknzhu.wixsite.com/mbweb`.

the authors, we use the *combine-skip model* that concatenates the vectors encoded by a uni-directional encoder (uni-skip) and a bi-directional encoder (bi-skip). The resulting vectors are of 4,800 dimensions—the first 2,400 dimensions is the uni-skip vector, and the last 2,400 dimensions is the bi-skip vector. We learn a ME classifier based on this representation. This model has a total of $4{,}800 \times 5 = 24{,}000$ parameters.

(f) **B-GRU:** This is a variation of our B-LSTM$_{\text{conv-glove}}$ model, where we replace the LSTM cells with GRU cells (Cho et al. 2014) in the recurrent layer. This model has a total of 2 (bi-direction) $\times$ 3 (gates) $\times$ $\big(128^2$ (hidden-hidden) $+ 300 \times$ 128 (input-hidden)$\big) + 256 \times 5 = 329{,}984$ trainable parameters (excluding the biases). Our LSTM-based RNN model uses four gates, which gives a total of 439,552 parameters to train.

We notice that all these models have a large number of parameters to learn an effective classification model for our task using the sentence representation as input features. Similar to our B-LSTM, the B-GRU and the skip-thought models are compositional, that is, they compose the sentence representation from the representation of its words using the sentence structure. Although the 4,800 dimensional sentence representation for skip-thought is not learned on the task, the associated weight parameters in the ME$_{\text{skip-thought}}$ model are trained on the task.

Table 12 presents the results. It can be observed that in general the compositional methods perform better than the non-compositional ones (e.g., averaging, concatenation), and when the compositional method is trained on the task, we get the best performance on two out of three data sets. In particular, our B-LSTM$_{\text{conv-glove}}$ gets the best results on QC3 and TA, outperforming B-GRU$_{\text{conv-glove}}$ by a slight margin in F_1.[17] The ME$_{\text{skip-thought}}$ performs the best on BC3, and close to the best results on TA. This is not so surprising because the skip-thought model encodes a sentence like a neural conversation model (Vinyals and Le 2015), and it has been shown that such models capture information relevant to speech acts (Ritter, Cherry, and Dolan 2010).

To further analyze the cases where B-LSTM$_{\text{conv-glove}}$ makes a difference, Figure 7 shows the corresponding confusion matrices for B-LSTM$_{\text{conv-glove}}$ and MLP$_{\text{conv-glove}}$ on the concatenated testsets of QC3, TA, and BC3. In general, our classifiers get confused between Response and Statement, and between Suggestion and Statement the most. We noticed a similar observation in the human annotations, where annotators had difficulties with these three acts. It is noticeable that B-LSTM$_{\text{conv-glove}}$ is less affected by class imbalance, and it can detect the *Suggestion* and *Polite* acts much more correctly than MLP$_{\text{conv-glove}}$. This indicates that LSTM-RNNs can model the grammar of the sentence when composing the words into phrases and sentences sequentially.

5.2.4 Effectiveness of Domain Adaptation. We have seen that semi-supervised learning in the form of word embeddings learned from a large unlabeled conversational corpus benefits our B-LSTM model. In the previous section, we witnessed further performance gains by exploiting more labeled data from the synchronous domain (MRDA). However, these methods make a simplified assumption that the conversational data comes from the same distribution. As mentioned before, the conversations in QC3, TA, or BC3 are quite different from MRDA meeting conversations in terms of style (spoken vs. written)

17 There is no significant difference between the accuracy numbers for B-GRU and B-LSTM.

Table 12
Comparison of different sentence encoders on the concatenated (CONCAT) data set. Best results are **boldfaced**. Accuracies significantly higher than $ME_{skip\text{-}thought}$ are marked with *.

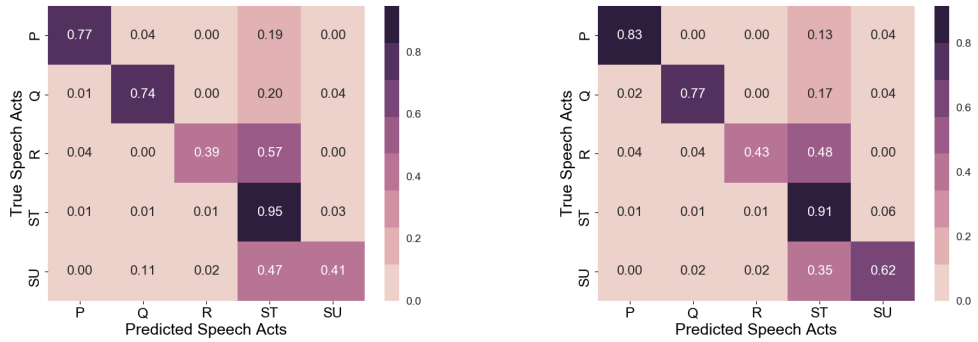
Encoder	Classifier	Model Name	QC3 (Testset)	TA (Testset)	BC3 (Testset)
Concatenation	ME	$ME_{conv\text{-}glove}$	60.52 (76.28)	75.47 (86.79)	60.46 (79.00)
Concatenation	MLP	$MLP_{conv\text{-}glove}$	60.47 (73.07)	75.85 (86.52)	55.33 (78.00)
Averaging	ME	$ME_{conv\text{-}glove\text{-}averaging}$	63.32 (76.92)	73.72 (84.09)	45.65 (74.00)
Averaging	SVM	$SVM_{conv\text{-}glove\text{-}averaging}$	18.74 (60.89)	29.46 (64.69)	16.19 (68.00)
Skip-thought	ME	$ME_{skip\text{-}thought}$	59.65 (78.13)	77.09 (86.22)	**71.89 (89.00)**
B-GRU	ME	$B\text{-}GRU_{conv\text{-}glove}$	69.45 **(81.41*)**	77.77 (88.68*)	58.66 (79.00)
B-LSTM	ME	$B\text{-}LSTM_{conv\text{-}glove}$	**70.51** (80.77*)	**78.08 (88.95*)**	58.28 (79.00)

and vocabulary usage. We believe that the results can be improved further by modeling the shift of domains (or distributions) explicitly.

In Section 3.2, we described two adaptation scenarios: (i) *unsupervised*, where no annotated data is available in the target domains, and (ii) *supervised*, where some annotated data is available in the target domain. We use all the available labels in the CONCAT data set for our supervised training. This makes the adaptation results comparable with our pre-adaptation results reported earlier in Table 12.

Table 13 presents the results for the adapted $B\text{-}LSTM_{conv\text{-}glove}$ model under the above training conditions (last two rows). For comparison, we have also shown the results for two baselines: (i) a *transfer* $B\text{-}LSTM_{conv\text{-}glove}$ model in the first row that is trained on only MRDA (source domain) data, and (ii) a *merge* $B\text{-}LSTM_{conv\text{-}glove}$ model in the second row that is trained on the concatenation of MRDA and the target domain data (QC3, TA, or BC3). Recall that the *merge* model is the one that gave the best results so far (i.e., last row of Table 11).

We can observe that without any labeled data in the target domain, the adapted $B\text{-}LSTM_{conv\text{-}glove}$ in the third row performs worse than the transfer baseline in QC3 and TA. In this case, because the out-of-domain labeled data set (MRDA) is much larger, it overwhelms the model, inducing features that are not relevant for the task in the target domain. However, when we provide the model with some labeled in-domain examples



(a) $MLP_{conv\text{-}glove}$ (b) $B\text{-}LSTM_{conv\text{-}glove}$

Figure 7
Confusion matrices for (a) $MLP_{conv\text{-}glove}$ and (b) $B\text{-}LSTM_{conv\text{-}glove}$ on the test sets of QC3, TA, and BC3. P stands for Polite, Q for Question, R for Response, ST for Statement, and SU stands for Suggestion.

Table 13
Results on the concatenated (CONCAT) data set with adversarial training.

	Training Regime	QC3 (Testset)	TA (Testset)	BC3 (Testset)
B-LSTM$_{\text{conv-glove}}$	Transfer	60.81 (72.35)	70.26 (83.85)	36.57 (57.14)
B-LSTM$_{\text{conv-glove}}$	Concatenation/Merge	70.51 (80.77)	78.08 (88.95)	58.28 (79.00)
Adapted B-LSTM$_{\text{conv-glove}}$	Unsupervised adaptation	47.67 (70.51)	66.17 (81.85)	39.97 (67.42)
Adapted B-LSTM$_{\text{conv-glove}}$	Supervised adaptation	**71.52 (82.48)**	**81.21 (88.67)**	**69.30 (88.22)**

in the supervised adaptation setting (last row), we observe gains over the merge model (second row) in all three data sets. Remarkably, the absolute improvements in F_1 for BC3 and TA are more than 11% and 3%, respectively.

To analyze further the performance of our adapted model, Figure 8 presents the F_1 scores with varying amounts of labeled data in the target domain. It can be noticed that for all three data sets, the largest improvements come from the first 25% of the labeled data. The gains from the second quartile are also relatively higher than the last two quartiles for TA and QC3. This demonstrates the effectiveness of our adversarial domain adaptation method. In the future, it will be interesting to compare adversarial training with other domain adaptation methods.

5.3 Effectiveness of CRFs

Conversation-Level Data Set for CRFs. To demonstrate the effectiveness of CRFs in capturing inter-sentence dependencies in an asynchronous conversation, we create another training setting called **CONV-LEVEL** (Conversation-level), in which training instances are entire conversations and the random splits are done at the *conversation* level (as opposed to sentence) for the asynchronous corpora. This is required because the CRFs perform joint learning and inference based on an entire

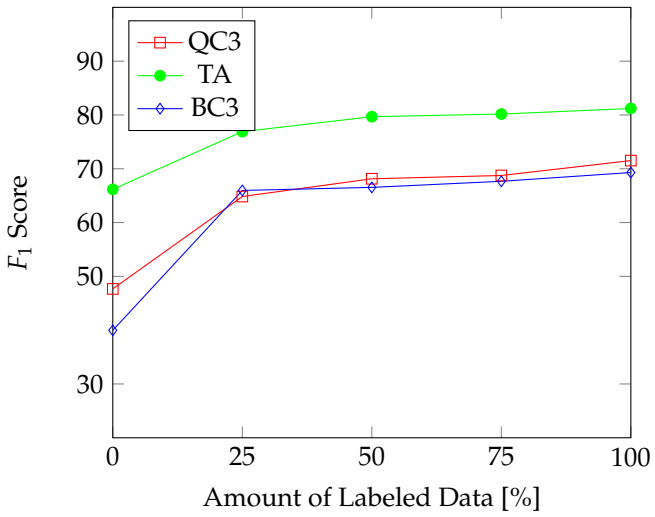


Figure 8
F_1 scores of our adapted model with varying amounts of labeled in-domain data.

Table 14
Data sets for CONV-LEVEL (conversation-level) setting to train, validate, and test our CRF models. Numbers in parentheses indicate the number of sentences.

	Train	Dev.	Test
QC3	38 (1,332)	4 (111)	5 (122)
TA	160 (2,957)	20 (310)	20 (444)
BC3	31 (1,012)	3 (101)	5 (222)
Total	229 (5,301)	27 (522)	30 (788)

conversation. Table 14 shows the resulting data sets that we use to train and evaluate our CRFs. We have 229 conversations for training and 27 conversations for development.[18] The test sets contain 5, 20, and 5 conversations for QC3, TA, and BC3, respectively.

Baselines and CRF Variants. We use the following three models as baselines:

(a) **$\mathbf{ME_{bow}}$**: a MaxEnt model with BOW representation.

(b) **Adapted B-$\mathbf{LSTM_{conv\text{-}glove}}$** (semi-supervised): This model performs adversarial semi-supervised domain adaptation using labeled sentences from MRDA and CONV-LEVEL training sets. Note that this is our best system so far (see Table 13).

(c) **$\mathbf{ME_{adapt\text{-}lstm}}$**: a MaxEnt learned from the sentence embeddings extracted from the adapted B-$\text{LSTM}_{\text{conv-glove}}$ (semi-supervised), that is, the sentence embeddings are used as feature vectors.

We experiment with the CRF variants shown in Table 1. Similar to $\text{ME}_{\text{adapt-lstm}}$, the CRFs are trained on the CONV-LEVEL training set using the sentence embeddings extracted by applying the adapted B-$\text{LSTM}_{\text{conv-glove}}$ (semi-supervised) model. The CRF models are therefore the structured versions of the $\text{ME}_{\text{adapt-lstm}}$ baseline.

Results and Discussion. Table 15 shows our results on the CONV-LEVEL data sets. We can notice that CRFs generally outperform MEs in accuracy, and for some CRF variants we get better results in both macro F_1 and accuracy. This indicates that there are conversational dependencies between the sentences in a conversation.

If we compare the CRF variants, we can see that the model that does not consider any link across comments ($\text{CRF}_{\text{LC-NO}}$) performs the worst. A simple linear chain connection between sentences in their temporal order ($\text{CRF}_{\text{LC-LC}}$) does not improve much. This indicates that the linear chain CRF (Lafferty, McCallum, and Pereira 2001) is not the most appropriate model for capturing conversational dependencies in asynchronous conversations.

The $\text{CRF}_{\text{LC-LC}_1}$ is one of the well performing models and performs significantly better than the adapted B-$\text{LSTM}_{\text{conv-glove}}$.[19] This model considers linear chain connections between sentences in a comment and only to the first comment. When we change this model to consider relations with every sentence in the first comment ($\text{CRF}_{\text{LC-FC}_1}$), this

18 We use the concatenated sets as train and dev sets.
19 Significance was computed based on the accuracy on the concatenated test set.

Table 15
Results of CRFs on CONV-LEVEL data set. Best results are **boldfaced**. Accuracies significantly higher than adapted B-LSTM$_{\text{conv-glove}}$ are marked with *.

	QC3	TA	BC3
ME$_{\text{bow}}$	57.37 (69.18)	65.39 (85.04)	60.32 (80.74)
Adapted B-LSTM$_{\text{conv-glove}}$ (semi-sup)	67.34 (80.15)	70.36 (86.73)	62.65 (83.59)
ME$_{\text{adapt-lstm}}$	62.36 (78.93)	63.31 (85.92)	58.32 (81.43)
CRF$_{\text{LC-NO}}$	67.02 (79.51)	67.82 (86.94)	63.15 (84.65)
CRF$_{\text{LC-LC}}$	67.12 (79.83)	67.94 (86.74)	63.75 (84.10)
CRF$_{\text{LC-LC}_1}$	69.32 **(81.03*)**	68.84 (87.34)	64.22 (84.71)
CRF$_{\text{LC-FC}_1}$	**70.11** (80.67)	69.73 (86.51)	**66.34 (86.51*)**
CRF$_{\text{FC-FC}}$	69.65 (80.77)	**72.31 (88.61*)**	64.82 (86.18*)

improves the performance further, giving the best results in two of the three data sets. This indicates that there are strong dependencies between the sentences of the initial comment and the sentences of the rest of the comments, and these dependencies are better captured if the relations between them are explicitly considered. The CRF$_{\text{FC-FC}}$ also yields as good results as CRF$_{\text{LC-FC}_1}$. This could be attributed to the robustness of the fully connected CRF, which learns from all possible pairwise relations.

Another interesting observation is that no single graph structure performs the best across all conversation types. For example, CRF$_{\text{LC-FC}_1}$ gives the highest F_1 scores for QC3 and BC3, whereas CRF$_{\text{FC-FC}}$ gives the highest results for TA. This shows the variation and the complicated ways in which participants communicate with each other in these conversations. One interesting future work would be to learn the underlying conversational structure automatically. However, we believe that in order to learn an effective model, this would require more labeled data.

To see some real examples in which CRF by means of its global learning and inference makes a difference, let us consider the example in Figure 1 again. We notice that the two sentences in comment C_4 were mistakenly identified as Statement and Response, respectively, by the B-LSTM$_{\text{conv-glove}}$ local model. However, by considering these two sentences together with others in the conversation, the global CRF$_{\text{LC-LC}_1}$, CRF$_{\text{LC-FC}_1}$, and CRF$_{\text{FC-FC}}$ models were able to correct them (see GLOBAL). CRF$_{\text{LC-LC}}$ could correctly identify the first one as Question.

6. Conclusions and Future Directions

We have presented a novel two-step framework for speech act recognition in asynchronous conversation. An LSTM-RNN first composes sentences of a conversation into vector representations by considering the word order in a sentence. Then a pairwise CRF jointly models the inter-sentence dependencies in a conversation. In order to mitigate the problem of limited annotated data in the asynchronous domains, we further adapt the LSTM-RNN to learn from synchronous meeting conversations using adversarial training of neural networks.

We experimented with different LSTM variants (uni- vs. bi-directional, random vs. pretrained initialization), and different CRF variants, depending on the underlying graph structure. We trained word2vec and Glove conversational word embeddings from a large conversational corpus. We trained our models on many different settings using synchronous and asynchronous corpora, including in-domain training,

concatenated training, unsupervised adaptation, supervised adaptation, and conversation level CRF joint training. We evaluated our approach on a synchronous data set (meeting) and three asynchronous data sets (two forum data sets and one e-mail data set), one of which is presented in this work.

Our experiments show that conversational word embeddings, especially conversational Glove, are quite beneficial for learning better sentence representations for the speech act classification task through bidirectional LSTM. This is especially true when the amount of labeled data in the target domain is limited. Adding more labeled data from synchronous domains yields improvements for bi-LSTMs, but even more gains can be achieved by domain adaptation with adversarial training. Further experiments with CRFs show that global joint models improve over local models given that the models consider the right graph structure. In particular, the LC-FC_1 and FC-FC graph structures were among the best performers.

This work leads us to a number of future directions. First, we would like to combine CRFs with LSTM-RNNs for doing the two steps jointly, so that the LSTM-RNNs can learn the embeddings directly using the global thread-level feedback. This would require the backpropagation algorithm to take error signals from the loopy BP inference. Second, we would also like to apply our models to conversations, where the graph structure is extractable using metadata or other clues, for example, the fragment quotation graphs for e-mail threads (Carenini, Ng, and Zhou 2008). One interesting future work would be to jointly model the conversational structure (e.g., reply-to structure) and the speech acts so that the two tasks can inform each other.

In another direction, we would like to evaluate our speech act recognition model on extrinsic tasks. In a separate thread of work, we are developing coherence models for asynchronous conversations (Nguyen and Joty 2017; Joty, Mohiuddin, and Tien Nguyen 2018). Such coherence models could be useful for a number of downstream tasks including next utterance (or comment) ranking, conversation generation, and thread reconstruction (Nguyen et al. 2017). We are now looking into whether speech act information can help us in building better coherence models for asynchronous conversations. We also plan to evaluate the utility of speech acts in downstream NLP tasks involving asynchronous conversations like next utterance ranking (Lowe et al. 2015), conversation generation (Ritter, Cherry, and Dolan 2010), and summarization (Murray, Carenini, and Ng 2010). Finally, we hope that the new corpus, the conversational word embeddings, and the source code that we have made publicly available in this work will facilitate other researchers in extending our work and in applying speech act models to their NLP tasks.

Bibliographic Note

Portions of this work were previously published in the ACL-2016 conference proceeding (Joty and Hoque 2016). This article significantly extends the published work in several ways, most notably: (i) we train new word2vec and Glove word embeddings based on a large conversational corpus, and show their effectiveness by comparing with off-the-shelf word embeddings (Section 4.2 and the Results section), (ii) we extend the LSTM-RNN for domain adaptation using adversarial training of neural networks (Section 3.2), (iii) we evaluate the domain adapted LSTM-RNN model on meeting and forum data sets (Section 5.2), and (iv) we train and evaluate CRFs based on sentence embeddings extracted from the adapted LSTM-RNN (Section 5.3). Beside these extensions, a significant portion of the article was rewritten to adapt to a journal-style publication.

Acknowledgments
We thank Aseel Ghazal for her efforts in creating the new QC3 corpus. We also thank Enamul Hoque for running and organizing some of the experiments reported in the ACL-2016 paper. Many thanks to the anonymous reviewers for their insightful comments on the ACL-2016 submitted version.

References

Allen, James, Nathanael Chambers, George Ferguson, Lucian Galescu, Hyuckchul Jung, Mary Swift, and William Taysom. 2007. Plow: A collaborative task learning agent. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 2*, AAAI'07, pages 1514–1519.

Arjovsky, Martín, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. *CoRR*, abs/1701.07875.

Austin, John Langshaw. 1962. *How To Do Things with Words*. Harvard University Press.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*, San Diego, CA.

Bangalore, Srinivas, Giuseppe Di Fabbrizio, and Amanda Stent. 2006. Learning the structure of task-driven human-human dialogs. In *Proceedings of the 44nd Annual Meeting of the Association for Computational Linguistics, ACL'06*, pages 201–208.

Bhatia, Sumit, Prakhar Biyani, and Prasenjit Mitra. 2014. Summarizing online forum discussions—can dialog acts of individual messages help? In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2127–2131, Doha.

Carenini, Giuseppe, Raymond T. Ng, and Xiaodong Zhou. 2008. Summarizing emails with conversational cohesion and subjectivity. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, ACL'08*, pages 353–361, OH.

Carvalho, Vitor R. and William W. Cohen. 2005. On the collective classification of e-mail "speech acts." In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 345–352, New York, NY.

Cho, Kyunghyun, Bart van Merriënboer, Gulcehre Caglar, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha.

Cohen, William W., Vitor R. Carvalho, and Tom M. Mitchell. 2004. Learning to classify e-mail into "speech acts." In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 309–316.

Dhillon, Rajdip, Sonali Bhagat, Hannah Carvey, and Elizabeth Shriberg. 2004. Meeting recorder project: Dialog act labeling guide. Technical report, ICSI Tech., Berkeley, USA.

Dielmann, Alfred and Steve Renals. 2008. Recognition of dialogue acts in multiparty meetings using a switching DBN. *IEEE Transactions on Audio, Speech, and Language Processing*, 16:1303–1314.

Dyer, Chris, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 334–343.

Feng, Donghui, Erin Shaw, Jihie Kim, and Eduard Hovy. 2006. Learning to detect conversation focus of threaded discussions. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 208–215, Stroudsburg, PA.

Ferschke, Oliver, Iryna Gurevych, and Yevgen Chebotar. 2012. Behind the article: Recognizing dialog acts in Wikipedia talk pages. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 777–786, Avignon.

Finkel, J., A. Kleeman, and C. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, ACL'08, pages 959–967, Columbus, OH.

Ganin, Yaroslav, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural

networks. *Journal of Machine Learning Research*, 17(1):2096–2030.

Gimpel, Kevin, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 42–47.

Glorot, Xavier and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *JMLR W&CP: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*, volume 9, pages 249–256, Sardinia.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of Advances in Neural Information Processing Systems Conference 27*, NIPS '14, pages 2672–2680, Montréal.

Hochreiter, Sepp and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hong, Liangjie and Brian D. Davison. 2009. A classification-based approach to question answering in discussion boards. In *32nd Annual International ACM SIGIR Conference on Research and Development on Information Retrieval*, pages 171–178, Boston, MA.

Irsoy, Ozan and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 720–728, Doha.

Jeong, Minwoo, Chin-Yew Lin, and Gary Geunbae Lee. 2009. Semi-supervised speech act recognition in emails and forums. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1250–1259, Singapore.

Ji, Yangfeng, Gholamreza Haffari, and Jacob Eisenstein. 2016. A latent variable recurrent neural network for discourse-driven language models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 332–342.

Joty, Shafiq, Giuseppe Carenini, and Chin-Yew Lin. 2011. Unsupervised modeling of dialog acts in asynchronous conversations. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, IJCAI'11, pages 1–130, Barcelona.

Joty, Shafiq, Giuseppe Carenini, and Raymond T. Ng. 2013. Topic segmentation and labeling in asynchronous conversations. *Journal of Artificial Intelligence Research*, 47(1):521–573.

Joty, Shafiq and Enamul Hoque. 2016. Speech act modeling of written asynchronous conversations with task-specific embeddings and conditional structured models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL '16, pages 1746–1756, Berlin.

Joty, Shafiq, Muhammad Tasnim Mohiuddin, and Dat Tien Nguyen. 2018. Coherence modeling of asynchronous conversations: A neural entity grid approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 558–568.

Julia, Fatema N. and Khan M. Iftekharuddin. 2008. Dialog act classification using acoustic and discourse information of maptask data. *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1472–1479.

Jurafsky, Dan, Liz Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual, draft 13. Technical Report, University of Colorado at Boulder & +SRI International.

Kalchbrenner, Nal and Phil Blunsom. 2013. Recurrent convolutional neural networks for discourse compositionality. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 119–126.

Khanpour, Hamed, Nishitha Guntakandla, and Rodney Nielsen. 2016. Dialogue act classification in domain-independent conversations using a deep recurrent neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2012–2021.

Kingma, Diederik P. and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Kiros, Ryan, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler.

2015. Skip-thought vectors. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 3294–3302, Cambridge, MA.

Lafferty, John, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA.

Lample, Guillaume, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Lee, Ji Young and Franck Dernoncourt. 2016. Sequential short-text classification with recurrent and convolutional neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–520.

Li, Jiwei, Thang Luong, Dan Jurafsky, and Eduard Hovy. 2015. When are tree structures necessary for deep learning of representations? In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2304–2314, Lisbon.

Liu, Pengfei, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1433–1443, Lisbon.

Louis, Annie and Shay B. Cohen. 2015. Conversation trees: A grammar model for topic structure in forums. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1543–1553, Lisbon.

Lowe, Ryan, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu dialogue corpus: A large data set for research in unstructured multi-turn dialogue systems. *CoRR*, abs/1506.08909.

Ma, Xuezhe and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNS-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074.

McKeown, Kathleen, Lokesh Shrestha, and Owen Rambow. 2007. Using question-answer pairs in extractive summarization of e-mail conversations. In *CICLing*, pages 542–550.

Mikolov, Tomáš. 2012. *Statistical language models based on neural networks.* Ph.D. thesis, Brno University of Technology.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '13, pages 746–751, Atlanta, GA.

Murphy, Kevin. 2012. *Machine Learning A Probabilistic Perspective*. The MIT Press.

Murphy, Kevin P., Yair Weiss, and Michael I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, UAI'99, pages 467–475, Stockholm.

Murray, Gabriel, Giuseppe Carenini, and Raymond Ng. 2010. Interpretation and transformation for abstracting conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 894–902, Los Angeles, CA.

Murray, Gabriel, Steve Renals, Jean Carletta, and Johanna Moore. 2006. Incorporating speaker and discourse features into speech summarization. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 367–374, Stroudsburg, PA.

Nair, Vinod and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, ICML '10, pages 807–814, Haifa.

Nguyen, Dat and Shafiq Joty. 2017. A neural local coherence model. In *Proceedings of the*

55th Annual Meeting of the Association for Computational Linguistics, ACL'17, pages 1320–1330, Association for Computational Linguistics, Vancouver.

Nguyen, Dat, Shafiq Joty, Basma Boussaha, and Maarten Rijke. 2017. Thread reconstruction in conversational data using neural coherence models. In *Proceedings of the Neu-IR 2017 SIGIR Workshop on Neural Information Retrieval*, NeuIR'17, Tokyo.

Oya, Tatsuro and Giuseppe Carenini. 2014. Extractive summarization and dialogue act modeling on e-mail threads: An integrated probabilistic approach. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 133–140, Philadelphia, PA.

Padó, Sebastian. 2006. *User's guide to `sigf`: Significance testing by approximate randomisation*.

Paul, Michael J. 2012. Mixed membership Markov models for unsupervised conversation modeling. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 94–104, Stroudsburg, PA.

Pearl, Judea. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP '14, pages 1532–1543, Doha.

Plank, Barbara, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418.

Qadir, Ashequl and Ellen Riloff. 2011. Classifying sentences as speech acts in message board posts. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 748–758, Edinburgh.

Ranganath, Rajesh, Dan Jurafsky, and Dan McFarland. 2009. It's not you, it's me: Detecting flirting and its misperception in speed-dates. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 334–342.

Ravi, Sujith and Jihie Kim. 2007. Profiling student interactions in threaded discussions with speech act classifiers. In *Proceedings of the 2007 Conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, pages 357–364, IOS Press, Amsterdam.

Ries, Klaus. 1999. HMM and neural network based speech act detection. In *ICASSP*, pages 497–500, Phoenix, AZ.

Ritter, Alan, Colin Cherry, and Bill Dolan. 2010. Unsupervised modeling of Twitter conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 172–180, Stroudsburg, PA.

Rush, Alexander M., Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Schegloff, Emanuel A. 1968. Sequencing in conversational openings. *American Anthropologist*, 70(6):1075–1095.

Schnabel, Tobias, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307, Lisbon.

Schuster, Mike and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Serban, Iulian V., Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 3776–3783.

Shen, Sheng-syun and Hung-yi Lee. 2016. Neural attention models for sequence classification: Analysis and application to key term extraction and dialogue act detection. *CoRR*, abs/1604.00077.

Shrestha, Lokesh and Kathleen McKeown. 2004. Detection of question-answer pairs in e-mail conversations. In *COLING '04: Proceedings of the 20th International Conference on Computational Linguistics*, page 889, Morristown, NJ.

Smith, Noah A. 2011. *Linguistic Structure Prediction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Stolcke, A., N. Coccaro, R. Bates, P. Taylor, C. Van Ess-Dykema, K. Ries, E. Shriberg, D. Jurafsky, R. Martin, and M. Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26:339–373.

Strubell, Emma, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '17, pages 2670–2680, Copenhagen.

Surendran, Dinoj and Gina-Anne Levow. 2006. Dialog act tagging with support vector machines and hidden markov models. In *Proceedings of Interspeech/ICSLP*, pages 1950–1953.

Sutton, C. and A. McCallum. 2012. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373.

Tavafi, Maryam, Yashar Mehdad, Shafiq Joty, Giuseppe Carenini, and Raymond Ng. 2013. Dialogue act recognition in synchronous and asynchronous conversations. In *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 117–121, Metz.

Tran, Quan Hung, Ingrid Zukerman, and Gholamreza Haffari. 2016. Inter-document contextual language model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 762–766.

Tran, Quan Hung, Ingrid Zukerman, and Gholamreza Haffari. 2017. Preserving distributional information in dialogue act classification. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2151–2156.

Turian, Joseph, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA.

Ulrich, Jan, Gabriel Murray, and Giuseppe Carenini. 2008. A publicly available annotated corpus for supervised e-mail summarization. In *AAAI'08 EMAIL Workshop*, pages 77–82, AAAI, Chicago, IL.

Vinyals, Oriol and Quoc V. Le. 2015. A neural conversational model. *CoRR*, abs/1506.05869.

Vishwanathan, S. V. N., Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. 2006. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 969–976, Pittsburgh, PA.

Vosoughi, Soroush and Deb Roy. 2016. Tweet acts: A speech act classifier for Twitter. In *Proceedings of the 10th International AAAI Conference on Weblogs and Social Media*, pages 711–714.

Weiss, Yair. 2001. Comparing the mean field method and belief propagation for approximate inference in MRFS. In Saad and Opper, editors, *Advanced Mean Field Methods*, pages 1–12, MIT Press.

Wilks, Yorick. 2006. Artificial companions as a new kind of interface to the future internet. *OII Research Report No. 13*.

Zhang, Renxian, Dehong Gao, and Wenjie Li. 2012. Towards scalable speech act recognition in Twitter: Tackling insufficient training data. In *Proceedings of the Workshop on Semantic Analysis in Social Media*, pages 18–27, Avignon.