Sentence Dependency Tagging in Online Question Answering Forums

Zhonghua Qu and **Yang Liu**
The University of Texas at Dallas
{qzh,yangl@hlt.utdallas.edu}

Abstract

Online forums are becoming a popular resource in the state of the art question answering (QA) systems. Because of its nature as an online community, it contains more updated knowledge than other places. However, going through tedious and redundant posts to look for answers could be very time consuming. Most prior work focused on extracting only question answering sentences from user conversations. In this paper, we introduce the task of sentence dependency tagging. Finding dependency structure can not only help find answer quickly but also allow users to trace back how the answer is concluded through user conversations. We use linear-chain conditional random fields (CRF) for sentence type tagging, and a 2D CRF to label the dependency relation between sentences. Our experimental results show that our proposed approach performs well for sentence dependency tagging. This dependency information can benefit other tasks such as thread ranking and answer summarization in online forums.

1 Introduction

Automatic Question Answering (QA) systems rely heavily on good sources of data that contain questions and answers. Question answering forums, such as technical support forums, are places where users find answers through conversations. Because of their nature as online communities, question answering forums provide more updated answers to new problems. For example, when the latest release of Linux has a bug, we can expect to find solutions in forums first. However, unlike other structured knowledge bases, often it is not straightforward to extract information such as questions and answers in online forums because such information spreads in the conversations among multiple users in a thread.

A lot of previous work has focused on extracting the question and answer sentences from forum threads. However, there is much richer information in forum conversations, and simply knowing a sentence is a question or answer is not enough. For example, in technical support forums, often it takes several iterations of asking and clarifications to describe the question. The same happens to answers. Usually several candidate answers are provided, and not all answers are useful. In this case users' feedback is needed to judge the correctness of answers.

Figure 1 shows an example thread in a technical support forum. Each sentence is labeled with its type (a detailed description of sentence types is provided Table 1). We can see from the example that questions and answers are not expressed in a single sentence or a single post. Only identifying question and answering sentences from the thread is not enough for automatic question answering. For this example, in order to get the complete question, we would need to know that sentence *S3* is a question that inquires for more details about the problem asked earlier, instead of stating its own question. Also, sentence *S5* should not be included in the correct answer since it is not a working solution, which is indicated by a negative feedback in sentence *S6*. The correct solution should be sentence *S7*, because of a user's positive confirmation *S9*. We define that there is a dependency between a pair of sentences if one sentence

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 554–562,
Jeju, Republic of Korea, 8-14 July 2012.

A: [S1:M-GRET] *Hi everyone.* **[S2:P-STAT]** *I have recently purchased USB flash and I am having trouble renaming it, please help me.*
B: [S3:A-INQU] *What is the size and brand of this flash?*
A: [S4:Q-CLRF] *It is a 4GB SanDisk flash.*
B: [S5:A-SOLU] *Install gparted, select flash drive and rename.*
A: [S6:M-NEGA] *I got to the Right click on partition and the label option was there but grayed out.*
B: [S7:A-SOLU] *Sorry again, I meant to right click the partition and select Unmount and then select Change name while in gparted.*
A: [S8:C-GRAT] *Thank you so much.* **[S9:M-POST]** *I now have an "Epic USB" You Rock!*

Figure 1: Example of a Question Answering Thread in Ubuntu Support Forum

exists as a result of another sentence. For example, question context sentences exist because of the question itself; an answering sentence exists because of a question; or a feedback sentence exists because of an answer. The sentence dependency structure of this example dialog is shown in Figure 2.

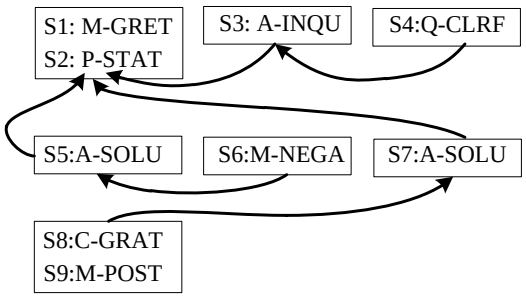

Figure 2: Dependency Structure of the Above Example

This example shows that in order to extract information from QA forums accurately, we need to understand the sentence dependency structure of a QA thread. Towards this goal, in this paper, we define two tasks: labeling the types for sentences, and finding the dependency relations between sentences.

For the first task of sentence type labeling, we define a rich set of categories representing the purpose of the sentences. We use linear-chain conditional random fields (CRF) to take advantage of many long-distance and non-local features. The second task is to identify relations between sentences. Most previous work only focused on finding the answer-question relationship between sentences. However, other relations can also be useful for information extraction from online threads, such as user's feedbacks on the answers, problem detail inquiry and question clarifications. In this study, we use two approaches for labeling of dependency relation between sentences. First each sentence is considered as a source, and we run a linear-chain CRF to label whether each of the other sentences is its target. Because multiple runs of separate linear-chain CRFs ignore the dependency between source sentences, the second approach we propose is to use a 2D CRF that models all pair relationships jointly.

The data we used was collected from Ubuntu forum general help section. Our experimental results show that our proposed sentence type tagging method works very well, even for the minority categories, and that using 2D CRF further improves performance over linear-chain CRFs for identifying dependency relation between sentences.

The paper is organized as follows. In the following section, we discuss related work on finding questions and answers in online environment as well as some dialog act tagging techniques. In Section 3, we introduce the use of CRFs for sentence type and dependency tagging. Section 4 describes data collection, annotation, and some analysis. In Section 5, we show that our approach achieves promising results in thread sentence dependency tagging. Finally we conclude the paper and suggest some possible future extensions.

2 Related Work

There is a lot of useful knowledge in the user generated content such as forums. This knowledge source could substantially help automatic question answering systems. There has been some previous work focusing on the extraction of question and corresponding answer pairs in online forums. In (Ding et al., 2008), a two-pass approach was used to find relevant solutions for a given question, and a skip-chain CRF was adopted to model long range de-

pendency between sentences. A graph propagation method was used in (Cong et al., 2008) to rank relevant answers to questions. An approach using email structure to detect and summarize question answer pairs was introduced in (Shrestha and Mckeown, 2004). These studies focused primarily on finding questions and answers in an online environment. In this paper, in order to provide a better foundation for question answer detection in online forums, we investigate tagging sentences with a much richer set of categories, as well as identifying their dependency relationships. The sentence types we use are similar to dialog acts (DA), but defined specifically for question answering forums. Work of (Clark and Popescu-Belis, 2004) defined a reusable multi-level tagset that can be mapped from conversational speech corpora such as the ICSI meeting data. However, it is hard to reuse any available corpus or DA tagset because our task is different, and also online forum has a different style from speech data. Automatic DA tagging has been studied a lot previously. For example, in (Stolcke et al., 2000), Hidden Markov Models (HMMs) were used for DA tagging; in (Ji and Bilmes, 2005), different types of graphical models were explored.

Our study is different in several aspects: we are using forum domains, unlike most work of DA tagging on conversational speech; we use CRFs for sentence type tagging; and more importantly, we also propose to use different CRFs for sentence relation detection. Unlike the pair-wise sentence analysis proposed in (Boyer et al., 2009) in which HMM was used to model the dialog structure, our model is more flexible and does not require related sentences to be adjacent.

3 Thread Structure Tagging

As described earlier, we decompose the structure analysis of QA threads into two tasks, first determine the sentence type, and then identify related sentences. This section provides details for each task.

3.1 Sentence Type Tagging

In human conversations, especially speech conversations, DAs have been used to represent the purpose or intention of a sentence. Different sets of DAs have been adopted in various studies, ranging from very coarse categories to fine grained ones. In this study, we define 13 fine grained sentence types (corresponding to 4 coarse categories) tailored to our domain of QA forum threads. Table 1 shows the categories and their description. Some tags such as P-STAT and A-SOLU are more important in that users try to state a problem and provide solutions accordingly. These are the typical ones used in previous work on question answering. Our set includes other useful tags. For example, C-NEGA and C-POSI can evaluate how good an answer is. Even though C-GRAT does not provide any direct feedback on the solutions, existence of such a tag often strongly implies a positive feedback to an answer. These sentence types can be grouped into 4 coarse categories, as shown in Table 1.

Types	Category	Description
Problems	P-STAT	question of problem
	P-CONT	problem context
	P-CLRF	problem clarification
Answers	A-SOLU	solution sentence
	A-EXPL	explanation on solutions
	A-INQU	inquire problem details
Confirm.	C-GRAT	gratitude
	C-NEGA	negative feedback
	C-POSI	positive feedback
Misc.	M-QCOM	question comment
	M-ACOM	comment on the answer
	M-GRET	greeting and politeness
	M-OFF	off-topic sentences

Table 1: Sentence Types for QA Threads

To automatically label sentences in a thread with their types, we adopt a sequence labeling approach, specifically linear-chain conditional random fields (CRFs), which have shown good performance in many other tasks (Lafferty, 2001). Intuitively there is a strong dependency between adjacent sentences. For example, in our data set, 45% sentences following a greeting sentence (M-GRET) are question related sentences; 53% sentences following a question inquiry sentence (Q-INQ) are solution related sentences. The following describes our modeling approaches and features used for sentence type tagging.

3.1.1 Linear-chain Conditional Random Field

Linear-chain CRFs is a type of undirected graphical models. Distribution of a set of variables in undirected graphical models can be written as

$$p(x, y) = \frac{1}{Z} \prod_{A} \psi_A(x_A, y_A) \quad (1)$$

Z is the normalization constant to guarantee valid probability distributions. CRFs is a special case of undirected graphical model in which ψ are log-linear functions:

$$\psi_A(x_A, y_A) = exp\left\{ \sum_k \theta_{A_k} f_{A_k}(x_A, y_A) \right\} \quad (2)$$

θ_A is a real value parameter vector for feature function set f_A. In the sequence labeling task, feature functions across the sequence are often tied together. In other words, feature functions at different locations of the sequence share the same parameter vector θ.

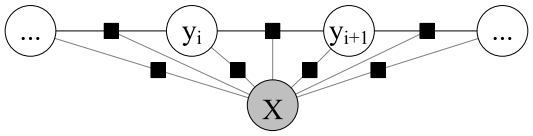

Figure 3: Graphical Structure of Linear-chain CRFs.

Linear-chain CRF is a special case of the general CRFs. In linear-chain CRF, cliques only involve two adjacent variables in the sequence. Figure 3 shows the graphical structure of a linear-chain CRF. In our case of sentence tagging, cliques only contain two adjacent sentences. Given the observation x, the probability of label sequence y is as follows:

$$p(y|x) = \frac{1}{Z} \prod_{i=1}^{|y|} \psi_e(x, y, i) \prod_{j=0}^{|y|} \psi_v(x, y, j) \quad (3)$$

$$\psi_e(x, y, i) = exp\left\{ \sum_k \theta_{e_k} f_{e_k}(y_{i-1}, y_i, x, i) \right\} \quad (4)$$

$$\psi_v(x, y, j) = exp\left\{ \sum_k \theta_{v_k} f_{v_k}(y_j, x, j) \right\} \quad (5)$$

where feature templates f_{e_k} and f_{v_k} correspond to edge features and node features respectively.

Feature Description
Cosine similarity with previous sentence.
Quote segment within two adjacent sentences?
Code segment within two adjacent sentences?
Does this sentence belong to author's post?
Is it the first sentence in a post?
Post author participated thread before?
Does the sentence contain any negative words?
Does the sentence contain any URL?
Does the sentence contain any positive words?
Does the sentence contain any question mark?
Length of the sentence.
Presence of verb.
Presence of adjective.
Sentence perplexity based on a background LM.
Bag of word features.

Table 2: Features Used in Sentence Type Tagging.

3.1.2 Sentence Type Tagging Features

We used various types of feature functions in sentence type tagging. Table 2 shows the complete list of features we used. Edge features are closely related to the transition between sentences. Here we use the cosine similarity between sentences, where each sentence is represented as a vector of words, with term weight calculated using TD-IDF (term frequency times inverse document frequency). High similarity between adjacent sentences suggests similar or related types. For node features, we explore different sources of information about the sentence. For example, the presence of a question mark indicates that a sentence may be a question or inquiry. Similarly, we include other cues, such as positive or negative words, verb and adjective words. Since technical forums tend to contain many system outputs, we include the perplexity of the sentence as a feature which is calculated based on a background language model (LM) learned from common English documents. We also use bag-of-word features as in many other text categorization tasks.

Furthermore, we add features to represent post level information to account for the structure of QA threads, for example, whether or not a sentence belongs to the author's post, or if a sentence is the beginning sentence of a post.

3.2 Sentence Dependency Tagging

Knowing only the sentence types without their dependency relations is not enough for question answering tasks. For example, correct labeling of an answer without knowing which question it actually refers to is problematic; not knowing which answer a positive or negative feedback refers to will not be helpful at all. In this section we describe how sentence dependency information is determined. Note that sentence dependency relations might not be a one-to-one relation. A many-to-many relation is also possible. Take question answer relation as an example. There could be potentially many answers spreading in many sentences, all depending on the same question. Also, it is very likely that a question is expressed in multiple sentences too.

Dependency relationship could happen between many different types of sentences, for example, answer(s) to question(s), problem clarification to question inquiry, feedback to solutions, etc. Instead of developing models for each dependency type, we treat them uniformly as dependency relations between sentences. Hence, for every two sentences, it becomes a binary classification problem, i.e., whether or not there exists a dependency relation between them. For a pair of sentences, we call the depending sentence the **source** sentence, and the depended sentence the **target** sentence. As described earlier, one source sentence can potentially depend on many different target sentences, and one target sentence can also correspond to multiple sources.

The sentence dependency task is formally defined as, given a set of sentences S_t of a thread, find the dependency relation $\{(s,t)|s \in S_t, t \in S_t\}$, where s is the source sentence and t is the target sentence that s depends on.

We propose two methods to find the dependency relationship. In the first approach, for each source sentence, we run a labeling procedure to find the dependent sentences. From the data, we found given a source sentence, there is strong dependency between adjacent target sentences. If one sentence is a target sentence of the source, often the next sentence is a target sentence too. In order to take advantage of such adjacent sentence dependency, we use the linear-chain CRFs for the sequence labeling. Features used in sentence dependency labeling are listed in Table 3. Note that a lot of the node features used here are relative to the source sentence since the task here is to determine if the two sentences are related. For a thread of N sentences, we need to perform N runs of CRF labeling, one for each sentence (as the source sentence) in order to label the target sentence corresponding to this source sentence.

	Feature Description
*	Cosine similarity with previous sentence.
*	Is adjacent sentence of the same type?
*	Pair of types of the adjacent target sentences.
	Pair of types of the source and target sentence.
	Is target in the same post as the source?
	Do target and source belong to the same author?
	Cosine similarity between target and source sentence.
	Does target sentence happen before source?
	Post distance between source and target sentence.

* indicates an edge feature

Table 3: Features Used in Sentence Dependency Labeling

The linear-chain CRFs can represent the dependency between adjacent target sentences quite well. However they cannot model the dependency between adjacent source sentences, because labeling is done for each source sentence individually. To model the dependency between both the source sentences and the target sentences, we propose to use 2D CRFs for sentence relation labeling. 2D CRFs are used in many applications considering two dimension dependencies such as object recognitions (Quattoni et al., 2004) and web information extraction (Zhu et al., 2005). The graphical structure of a 2D CRF is shown in Figure 4. Unlike one dimensional sequence labeling, a node in 2D environment is dependent on both x-axis neighbors and y-axis neighbors. In the sentence relation task, the source and target pair is a 2D relation in which its label depends on labels of both its adjacent source and its adjacent target sentence. As shown in Figure 4, looking from x-axis is the sequence of target sentences with a fixed source sentence, and from y-axis is the sequence of source sentences with a fixed target sentence. This model allows us to model all the sentence relationships jointly. 2D CRFs contain 3 templates of features: node template, x-axis edge template, and y-axis edge template. We use the same edge features and node features as in linear-chain CRFs for node features and y-axis edge features in

2D CRFs. For the x-axis edge features, we use the same feature functions as for y-axis, except that now they represent the relation between adjacent source sentences.

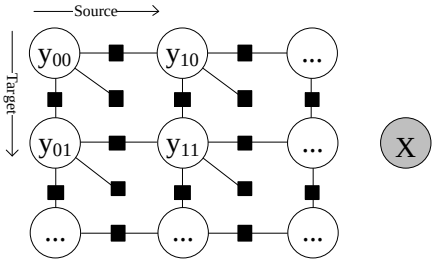

Figure 4: Graphical Structure of 2D CRF for Sentence Relation Labeling.

In a thread containing N sentences, we would have a 2D CRF containing N^2 nodes in a $N \times N$ grid. Exact inference in such a graph is intractable. In this paper we use loopy belief propagation algorithm for the inference. Loopy belief propagation is a message passing algorithm for graph inference. It calculates the marginal distribution for each node in the graph. The result is exact in some graph structures (e.g., linear-chain CRFs), and often converges to a good approximation for general graphs.

4 Data

We used data from ubuntu community forum general help section for the experiments and evaluation. This is a technical support section that provides answers to the latest problems in Ubuntu Linux. Among all the threads that we have crawled, we selected 200 threads for this initial study. They contain between $2-10$ posts and at least 2 participants. Sentences inside each thread are segmented using Apache OpenNLP tools. In total, there are 706 posts and 3,483 sentences. On average, each thread contains 3.53 posts, and each post contains around 4.93 sentences. Two annotators were recruited to annotate the sentence type and the dependency relation between sentences. Annotators are both computer science department undergraduate students. They are provided with detailed explanation of the annotation standard. The distribution of sentence types in the annotated data is shown in Table 4, along with inter-annotator Kappa statistics calculated using 20 common threads annotated by both annotators. We can see that the majority of the sentences are about problem descriptions and solutions. In general, the agreement between the two annotators is quite good.

General Type	Category	Percentage	Kappa
Problems	P-STAT	12.37	0.88
	P-CONT	37.30	0.77
	P-CLRF	1.01	0.98
Answers	A-SOLU	9.94	0.89
	A-EXPL	11.60	0.89
	A-INQU	1.38	0.99
Confirmation	C-GRAT	5.06	0.98
	C-NEGA	1.98	0.96
	C-POSI	1.84	0.96
Miscellaneous	M-QCOM	1.98	0.93
	M-ACOM	1.47	0.96
	M-GRET	1.01	0.96
	M-OFF	7.92	0.96

Table 4: Distribution and Inter-annotator Agreement of Sentence Types in Data

There are in total $1,751$ dependency relations identified by the annotators among those tagged sentences. Note that we are only dealing with intra-thread sentence dependency, that is, no dependency among sentences in different threads is labeled. Considering all the possible sentence pairs in each thread, the labeled dependency relations represent a small percentage. The most common dependency is problem description to problem question. This shows that users tend to provide many details of the problem. This is especially true in technical forums. Seeing questions without their context would be confusing and hard to solve. The relation of answering solutions and question dependency is also very common, as expected. The third common relation is the feedback dependency. Even though the number of feedback sentences is small in the data set, it plays a vital role to determine the quality of answers. The main reason for the small number is that, unlike problem descriptions, much fewer sentences are needed to give feedbacks.

5 Experiment

In the experiment, we randomly split annotated threads into three disjoint sets, and run a three-fold cross validation. Within each fold, first sentence types are labeled using linear-chain CRFs, then the

resulting sentence type tagging is used in the second pass to determine dependency relations. For part-of-speech (POS) tagging of the sentences, we used Stanford POS Tagger (Toutanova and Manning, 2000). All the graphical inference and estimations are done using MALLET package (McCallum, 2002).

In this paper, we evaluate the results using standard precision and recall. In the sentence type tagging task, we calculate precision, recall, and F_1 score for each individual tag. For the dependency tagging task, a pair identified by the system is correct only if the exact pair appears in the reference annotation. Precision and recall scores are calculated accordingly.

5.1 Sentence Type Tagging Results

The results of sentence type tagging using linear-chain CRFs are shown in Table 5. For a comparison, we include results using a basic first-order HMM model. Because HMM is a generative model, we use only bag of word features in the generative process. The observation probability is the probability of the sentence generated by a unigram language model, trained for different sentence types. Since for some applications, fine grained categories may not be needed, for example, in the case of finding questions and answers in a thread, we also include in the table the tagging results when only the general categories are used in both training and testing.

We can see from the table that using CRFs achieves significantly better performance than HMMs for most categories, except greeting and off-topic types. This is mainly because of the advantage of CRFs, allowing the incorporation of rich discriminative features. For the two major types of problems and answers, in general, our system shows very good performance. Even for minority types like feedbacks, it also performs reasonably well. When using coarse types, the performance on average is better compared to the finer grained categories, mainly because of the fewer classes in the classification task. Using the fine grained categories, we found that the system is able to tell the difference between "problem statement" (P-STAT) and "problem context" (P-CONT). Note that in our task, a problem statement is not necessarily a question sentence. Instead it could be any sentence that expresses the need for a solution.

	Linear-chain CRF		First-order HMM	
13 Fine Grained Types				
Tag	**Prec. / Rec.**	F_1	**Prec. / Rec.**	F_1
M-GRET	0.45 / 0.58	0.51	0.73 / 0.57	**0.64**
P-STAT	0.79 / 0.72	**0.75**	0.35 / 0.34	0.35
P-CONT	0.80 / 0.74	**0.77**	0.58 / 0.18	0.27
A-INQU	0.37 / 0.48	**0.42**	0.11 / 0.25	0.15
A-SOLU	0.78 / 0.64	**0.71**	0.27 / 0.29	0.28
A-EXPL	0.4 / 0.76	**0.53**	0.24 / 0.19	0.21
M-POST	0.5 / 0.41	**0.45**	0.04 / 0.1	0.05
C-GRAT	0.43 / 0.53	**0.48**	0.01 / 0.25	0.02
M-NEGA	0.67 / 0.5	**0.57**	0.09 / 0.31	0.14
M-OFF	0.11 / 0.23	0.15	0.20 / 0.23	**0.21**
P-CLRF	0.15 / 0.33	**0.21**	0.10 / 0.12	0.11
M-ACOM	0.27 / 0.38	**0.32**	0.09 / 0.1	0.09
M-QCOM	0.34 / 0.32	**0.33**	0.08 / 0.23	0.11
4 General Types				
Tag	**Prec. / Rec.**	F_1	**Prec. / Rec.**	F_1
Problem	0.85 / 0.76	**0.80**	0.73 / 0.27	0.39
Answers	0.65 / 0.72	**0.68**	0.45 / 0.36	0.40
Confirm.	0.80 / 0.74	**0.77**	0.06 / 0.26	0.10
Misc.	0.43 / 0.61	**0.51**	0.04 / 0.36	0.08

Table 5: Sentence Type Tagging Performance Using CRFs and HMM.

We also performed some analysis of the features using the feature weights in the trained CRF models. We find that some post level information is relatively important. For example, the feature representing whether the sentence is before a "code" segment has a high weight for problem description classification. This is because in linux support forum, people usually put some machine output after their problem description. We also notice that the weights for verb words are usually high. This is intuitive since the "verb" of a sentence can often determine its purpose.

5.2 Sentence Dependency Tagging Results

Table 6 shows the results using linear-chain CRFs (L-CRF) and 2D CRFs for sentence dependency tagging. We use different settings in our experiments. For the categories of sentence types, we evaluate using both the fine grained (13 types) and the coarse categories (4 types). Furthermore, we examine two ways to obtain the sentence types. First, we use the output from automatic sentence type tagging. In the second one, we use the sentence type information from the human annotated data in order to avoid the error propagation from automatic sentence type la-

beling. This gives an oracle upper bound for the second pass performance.

Using Oracle Sentence Type				
Setup		**Precision**	**Recall**	F_1
13 types	L-CRF	0.973	0.453	0.618
	2D-CRF	0.985	0.532	0.691
4 general	L-CRF	0.941	0.124	0.218
	2D-CRF	0.956	0.145	0.252
Using System Sentence Type				
Setup		**Precision**	**Recall**	F_1
13 types	L-CRF	0.943	0.362	0.523
	2D-CRF	0.973	0.394	0.561
4 general	L-CRF	0.939	0.101	0.182
	2D-CRF	0.942	0.127	0.223

Table 6: Sentence Dependency Tagging Performance

From the results we can see that 2D CRFs outperform linear-chain CRFs for all the conditions. This shows that by modeling the 2D dependency in source and target sentences, system performance is improved. For the sentence types, when using automatic sentence type tagging systems, there is a performance drop. The performance gap between using the reference and automatic sentence types suggests that there is still room for improvement from better sentence type tagging. Regarding the categories used for the sentence types, we observe that they have an impact on dependence tagging performance. When using general categories, the performance is far behind that using the fine grained types. This is because some important information is lost when grouping categories. For example, a dependency relation can be: "A-EXPL" (explanation for solutions) depends on "A-SOLU" (solutions); however, when using coarse categories, both are mapped to "Solution", and having one "Solution" depending on another "Solution" is not very intuitive and hard to model properly. This shows that detailed category information is very important for dependency tagging even though the tagging accuracy from the first pass is far from perfect.

Currently our system does not put constraints on the sentence types for which dependencies exist. In the system output we find that sometimes there are obvious dependency errors, such as a positive feedback depending on a negative feedback. We may improve our models by taking into account different sentence types and dependency relations.

6 Conclusion

In this paper, we investigated sentence dependency tagging of question and answer (QA) threads in online forums. We define the thread tagging task as a two-step process. In the first step, sentence types are labeled. We defined 13 sentence types in order to capture rich information of sentences to benefit question answering systems. Linear chain CRF is used for sentence type tagging. In the second step, we label actual dependency between sentences. First, we propose to use a linear-chain CRF to label possible target sentences for each source sentence. Then we improve the model to consider the dependency between sentences along two dimensions using a 2D CRF. Our experiments show promising performance in both tasks. This provides a good pre-processing step towards automatic question answering. In the future, we plan to explore using constrained CRF for more accurate dependency tagging. We will also use the result from this work in other tasks such as answer quality ranking and answer summarization.

7 Acknowledgment

This work is supported by DARPA under Contract No. HR0011-12-C-0016 and NSF No. 0845484. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of DARPA or NSF.

References

Kristy Elizabeth Boyer, Robert Phillips, Eun Young Ha, Michael D. Wallis, Mladen A. Vouk, and James C. Lester. 2009. Modeling dialogue structure with adjacency pair analysis and hidden markov models. In *Proc. NAACL-Short*, pages 49–52.

Alexander Clark and Andrei Popescu-Belis. 2004. Multi-level dialogue act tags. In *Proc. SIGDIAL*, pages 163–170.

Gao Cong, Long Wang, Chinyew Lin, Youngin Song, and Yueheng Sun. 2008. Finding question-answer pairs from online forums. In *Proc. SIGIR*, pages 467–474.

Shilin Ding, Gao Cong, Chinyew Lin, and Xiaoyan Zhu. 2008. Using conditional random fields to extract contexts and answers of questions from online forums. In *Proc. ACL-HLT*.

Gang Ji and J Bilmes. 2005. Dialog Act Tagging Using Graphical Models. In *Proc. ICASSP*.

John Lafferty. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, pages 282–289.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. *http://mallet.cs.umass.edu*.

Ariadna Quattoni, Michael Collins, and Trevor Darrell. 2004. Conditional random fields for object recognition. In *Proc. NIPS*, pages 1097–1104.

Lokesh Shrestha and Kathleen Mckeown. 2004. Detection of question-answer pairs in email conversations. In *Proc. Coling*, pages 889–895.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26:339–373.

Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proc. EMNLP/VLC*, pages 63–70.

Jun Zhu, Zaiqing Nie, Ji R. Wen, Bo Zhang, and Wei Y. Ma. 2005. 2D Conditional Random Fields for Web information extraction. In *Proc. ICML*, pages 1044–1051, New York, NY, USA.