A Structural Support Vector Method for Extracting Contexts and Answers of Questions from Online Forums

Wen-Yun Yang [†*] **Yunbo Cao** [†‡] **Chin-Yew Lin** [‡]
[†] Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
[‡] Microsoft Research Asia, Beijing, China
wenyun.yang@gmail.com {yunbo.cao; cyl}@microsoft.com

Abstract

This paper addresses the issue of extracting contexts and answers of questions from post discussion of online forums. We propose a novel and unified model by customizing the structural Support Vector Machine method. Our customization has several attractive properties: (1) it gives a comprehensive graphical representation of thread discussion. (2) It designs special inference algorithms instead of general-purpose ones. (3) It can be readily extended to different task preferences by varying loss functions. Experimental results on a real data set show that our methods are both promising and flexible.

Post1: <context id=1> **S1**: *Hi I am looking for a pet friendly hotel in Hong Kong because all of my family is going there for vacation.* **S2**: *my family has 2 sons and a dog.* </context> <question id=1> **S3**: *Is there any recommended hotel near Sheung Wan or Tsing Sha Tsui?* </question> <context id=2, 3> **S4**: *We also plan to go shopping in Causeway Bay.* </context> <question id=2> **S5**: *What's the traffic situation around those commercial areas?* </question> <question id=3> **S6**: *Is it necessary to take a taxi?* </question> **S7**: *Any information would be appreciated.*
Post2: <answer id=1> **S8**: *The Comfort Lodge near Kowloon Park allows pet as I know, and usually fits well within normal budgets.* **S9**: *It is also conveniently located, nearby the Kowloon railway station and subway.* </answer>
Post3: <answer id=2, 3> **S10**: *It's very crowd in those areas, so I recommend MTR in Causeway Bay because it is cheap to take you around.* </answer>

Figure 1: An example thread with three posts and ten sentences

1 Introduction

Recently, extracting questions, contexts and answers from post discussions of online forums incurs increasing academic attention (Cong et al., 2008; Ding et al., 2008). The extracted knowledge can be used either to enrich the knowledge base of community question answering (QA) services such as Yahoo! Answers or to augment the knowledge base of chatbot (Huang et al., 2007).

Figure 1 gives an example of a forum thread with questions, contexts and answers annotated. This thread contains *three posts* and *ten sentences*, among which *three questions* are discussed. The three questions are proposed in three sentences, S3, S5 and S6. The *context sentences* S1 and S2 provide contextual information for question sentence S3. Similarly, the *context sentence* S4 provides contextual information for question sentence S5 and S6. There are three question-context-answer triples in this example, (S3) – (S1, S2) – (S8, S9), (S5) – (S4) – (S10) and (S6) – (S4) – (S10). As shown in the example, a forum question usually requires contextual information to complement its expression. For example, the question sentence S3 would be of incomplete meaning without the contexts S1 and S2, since the important keyword *pet friendly* would be lost.

The problem of extracting questions, contexts, and answers can be solved in two steps: (1) identify questions and then (2) extract contexts and answers for them. Since identifying questions from forum discussions is already well solved in (Cong et al., 2008), in this paper, we are focused on step (2) while assuming questions already identified.

Previously, Ding et al. (2008) employ general-purpose graphical models without any customizations to the specific extraction problem (step 2). In this paper, we improve the existing models in

[*] This work was done while the first author visited Microsoft Research Asia.

three aspects: *graphical representation*, *inference algorithm* and *loss function*.

Graphical representation. We propose a more comprehensive and unified graphical representation to model the thread for relational learning. Our graphical representation has two advantages over previous work (Ding et al., 2008): *unifying sentence relations* and *incorporating question interactions*.

Three types of relation should be considered for context and answer extraction: (a) relations between successive sentences (e.g., *context* sentence S2 occurs immediately before *question* sentence S3); (b) relations between *context* sentences and *answer* sentences (e.g., *context* S4 presents the phrase *Causeway Bay* linking to *answer* which is absent from *question* S6); and (c) relations between multiple labels for one sentence (e.g., one *question* sentence is unlikely to be the *answer* to another *question* although one sentence can serve as *contexts* for more than one *questions*). Our proposed graphical representation improves the modeling of the three types of sentence relation (Section 2.2).

Certain interactions exist among questions. For example, *question* sentences S5 and S6 interact by sharing *context* sentence S4. Our proposed graphical representation can naturally model the interactions. Previous work (Ding et al., 2008) performs the extraction of contexts and answers in multiple passes of the thread (with each pass corresponding to one question), which cannot address the interactions well. In comparison, our model performs the extraction in one pass of the thread.

Inference algorithm. Inference is usually a time-consuming process for structured prediction. We design special inference algorithms, instead of general-purpose inference algorithms used in previous works (Cong et al., 2008; Ding et al., 2008), by taking advantage of special properties of our task. Specifically, we utilize two special properties of thread structure to reduce the inference (time) cost. First, *context* sentences and *question* sentences usually occur in the same post while *answer* sentences can only occur in the following posts. With this properties, we can greatly reduce *context* (or *answer*) candidate sets of a question, which results in a significant decrease in inference cost (Section 3). Second, *context* candidate set is usually much smaller than the number of sentences in a thread. This property enables our proposal to have an exact and efficient inference (Section 4.1). Moreover, an approximate inference algorithm is also given (Section 4.2).

Loss function. In practice, different application settings usually imply different requirements for system performance. For example, we expect a higher recall for the purpose of archiving questions but a higher precision for the purpose of retrieving questions. A flexible framework should be able to cope with various requirements. We employ structural Support Vector Machine (SVM) model that could naturally incorporate different loss functions (Section 5).

We use a real data set to evaluate our approach to extracting contexts and answers of questions. The experimental results show both the effectiveness and the flexibility of our approach.

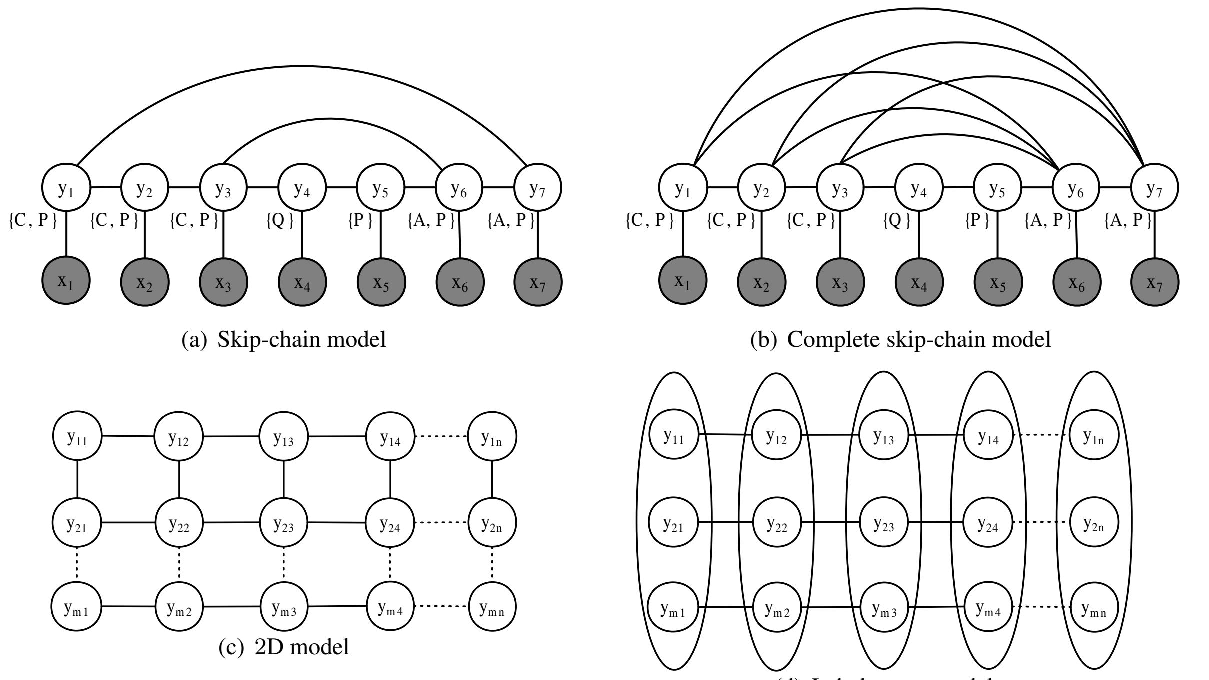
In the next section, we formalize the problem of context and answer extraction and introduce the structural model. In Sections 3, 4 and 5 we give the details of customizing structural model for our task. In Section 6, we evaluate our methods. In Section 7, we discuss the related work. Finally, we conclude this paper in Section 8.

2 Problem Statement

We first introduce our notations in Section 2.1 and then in Section 2.2 introduce how we model the problem of extracting contexts and answers for questions with a novel form of graphical representation. In Section 2.3 we introduce the structured model based on the new representation.

2.1 Notations

Assuming that a given thread contains p posts $\{p_1, \ldots, p_p\}$, which are authored by a set of users $\{u_1, \ldots, u_p\}$. The p posts can be further segmented into n sentences $\mathbf{x} = \{x_1, \ldots, x_n\}$. Among the n sentences, m *question* sentences $\mathbf{q} = \{x_{q_1}, \ldots, x_{q_m}\}$ have been identified. Our task is to identify the *context* sentences and the *answer* sentences for those m question sentences. More formally, we use four types of label $\{C, A, Q, P\}$ to stand for context, answer, question and plain labels. Then, our task is to predict an $m \times n$ label matrix $\mathbf{y} = (y_{ij})_{1 \leq i \leq m, 1 \leq j \leq n}$, except m elements $\{y_{1,q_1}, \ldots, y_{m,q_m}\}$ which correspond to (known) *question* labels. The element y_{ij} in label matrix $\mathbf{y}$ represents the role that the jth sentence plays for the ith question. We denote the ith row and jth column of the label matrix $\mathbf{y}$ by $\mathbf{y}_{i.}$ and $\mathbf{y}_{.j}$.



(a) Skip-chain model

(b) Complete skip-chain model

(c) 2D model

(d) Label group model

Figure 2: Structured models

2.2 Graphical Representation

Recently, Ding et al. (2008) use skip-chain and 2D Conditional Random Fields (CRFs) (Lafferty et al., 2001) to perform the relational learning for context and answer extraction. The skip-chain CRFs (Sutton and McCallum, 2004; Galley, 2006) model the long distance dependency between context and answer sentences and the 2D CRFs (Zhu et al., 2005) model the dependency between contiguous questions. The graphical representation of those two models are shown in Figures 2(a) and 2(c), respectively. Those two CRFs are both extensions of the linear chain CRFs for the sake of powerful relational learning. However, directly using the skip-chain and 2D CRFs without any customization has obvious disadvantages: (a) the skip-chain model does not model the *dependency between answer sentence and multiple context sentences*; and (b) the 2D model does not model the *dependency between non-contiguous questions*.

To better model the problem of extracting contexts and answers of questions, we propose two more comprehensive models, *complete skip-chain model* and *label group model* to improve the capability of the two previous models. These two models are shown in Figures 2(b) and 2(d).

In Figures 2(a) and 2(b), each label node is annotated with its allowed labels and the labels C, A, Q and P stand for context, answer, question and plain sentence labels, respectively. Note that the complete skip-chain model *completely* links each two context and answer candidates and the label group model combines the labels of one sentence into one *label group*.

2.3 Structured Model

Following the standard machine learning setup, we denote the input and output spaces by $\mathcal{X}$ and $\mathcal{Y}$, then formulate our task as learning a hypothesis function $h : \mathcal{X} \rightarrow \mathcal{Y}$ to predict a $\mathbf{y}$ when given $\mathbf{x}$. In this setup, $\mathbf{x}$ represents a thread of n sentences and m identified questions. $\mathbf{y}$ represents the $m \times n$ label matrix to be predicted.

Given a set of training examples, $S = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \in \mathcal{X} \times \mathcal{Y} : i = 1, \ldots, N\}$, we restrict ourselves to the supervised learning scenario. We focus on hypothesis functions that take the form $h(\mathbf{x}; \mathbf{w}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} \mathcal{F}(\mathbf{x}, \mathbf{y}; \mathbf{w})$ with discriminant function $\mathcal{F} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$ where $\mathcal{F}(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \mathbf{w}^T \Psi(\mathbf{x}, \mathbf{y})$. As will be introduced in Section 4, we employ structural SVMs (Joachims et al., 2009) to find the optimal parameters $\mathbf{w}$. The structural SVMs have several competitive properties as CRFs. First, it follows from the maximum margin strategy, which has been shown with competitive or even better performance (Tsochantaridis et al., 2005; Nguyen and Guo, 2007). Second, it allows flexible choices of loss functions to users. Moreover, in general, it has theoretically proved convergence in polynomial time (Joachims et al., 2009).

To use structural SVMs in relational learning, one needs to customize three steps according to specific tasks. The three steps are (a) *definition of joint feature mapping* for encoding relations, (b) *algorithm of finding the most violated constraint (inference)* for efficient trainings and (c) *definition of loss function* for flexible uses.

In the following Sections 3, 4 and 5, we describe the customizations of the three steps for our context and answer extraction task, respectively.

3 Encoding Relations

We use a joint feature mapping to model the relations between sentences in a thread. For context and answer extraction, the joint feature mapping can be defined as follows,

$$\Psi(\mathbf{x},\mathbf{y}) = \begin{bmatrix} \Psi_n(\mathbf{x},\mathbf{y}) \\ \Psi_h(\mathbf{x},\mathbf{y}) \\ \Psi_v(\mathbf{x},\mathbf{y}) \end{bmatrix},$$

where the sub-mappings $\Psi_n(\mathbf{x},\mathbf{y})$, $\Psi_h(\mathbf{x},\mathbf{y})$, and $\Psi_v(\mathbf{x},\mathbf{y})$ encode three types of feature mappings, *node features*, *edge features* and *label group features*. The node features provide the basic information for the output labels. The edge features consist of the sequential edge features and skip-chain edge features for successive label dependencies. The label group features encode the relations within each label group.

Before giving the detail definitions of the sub-mappings, we first introduce *the context and answer candidate sets*, which will be used for the definitions and inferences. Each row of the label matrix $\mathbf{y}$ corresponds to one question. Assuming that the ith row $\mathbf{y}_{i\cdot}$ corresponds to the question with sentence index q_i, we thus have two candidate sets of contexts and answers for this question denoted by $\mathcal{C}$ and $\mathcal{A}$, respectively. We denote the post indices and the author indices for the n sentences as $\mathbf{p} = (p_1, \ldots, p_n)$ and $\mathbf{u} = (u_1, \ldots, u_n)$. Then, we can formally define the two candidate sets for the $\mathbf{y}_{i\cdot}$ as

$$\mathcal{C} = \left\{ c_j \,\middle|\, \underbrace{p_{c_j} = p_{q_i}}_{\text{In Question Post}}, \underbrace{c_j \neq q_i}_{\text{Not Question Sentence}} \right\},$$

$$\mathcal{A} = \left\{ a_j \,\middle|\, \underbrace{p_{a_j} > p_{q_i}}_{\text{After Question Post}}, \underbrace{u_{a_j} \neq u_{q_i}}_{\text{Not by the Same User}} \right\}.$$

In the following, we describe formally about the definitions of the three feature sub-mappings.

The node feature mapping $\Psi_n(\mathbf{x},\mathbf{y})$ encodes the relations between sentence and label pairs, we define it as follows,

$$\Psi_n(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \psi_n(x_j, y_{ij}),$$

where $\psi_n(x_j, y_{ij})$ is a feature mapping for a given sentence and a label. It can be formally defined as follows,

$$\psi_n(x_j, y_{ij}) = \Lambda(y_{ij}) \otimes \phi_{q_i}(x_j), \tag{1}$$

where $\otimes$ denotes a tensor product, $\phi_{q_i}(x_j)$ and $\Lambda(y_{ij})$ denote two vectors. $\phi_{q_i}(x_j)$ contains basic information for output label. $\Lambda(y_{ij})$ is a $0/1$ vector defined as

$$\Lambda(y_{ij}) = [\lambda_C(y_{ij}), \lambda_A(y_{ij}), \lambda_P(y_{ij})]^T,$$

where $\lambda_C(y_{ij})$ equal to one if $y_{ij} = C$, otherwise zero. The $\lambda_A(y_{ij})$ and $\lambda_P(y_{ij})$ are similarly defined. Thus, for example, writing out $\psi_n(x_j, y_{ij})$ for $y_{ij} = C$ one gets,

$$\psi_n(x_j, y_{ij}) = \begin{pmatrix} \phi_{q_i}(x_j) \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \begin{matrix} \leftarrow & \text{context} \\ \leftarrow & \text{answer} \\ \leftarrow & \text{plain} \end{matrix}.$$

Note that the node feature mapping does not incorporate the relations between sentences.

The edge feature mapping $\Psi_h(\mathbf{x},\mathbf{y})$ is used to incorporate two types of relation, the relation between successive sentences and the relation between context and answer sentences. It can be defined as follows,

$$\Psi_h(\mathbf{x},\mathbf{y}) = \begin{bmatrix} \Psi_{hn}(\mathbf{x},\mathbf{y}) \\ \Psi_{hc}(\mathbf{x},\mathbf{y}) \end{bmatrix},$$

where $\Psi_{hn}(\mathbf{x},\mathbf{y})$ and $\Psi_{hc}(\mathbf{x},\mathbf{y})$ denote the two types of feature mappings corresponding to sequential edges and skip-chain edges, respectively. Their formal definitions are given as follows,

$$\Psi_{hn}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m}\sum_{j=1}^{n-1} \psi_{hn}(x_j, x_{j+1}, y_{ij}, y_{i,j+1}),$$

Descriptions	Dimensions
$\psi_{q_i}(x_j)$ (32 dimensions) in $\Psi_n(\mathbf{x},\mathbf{y})$	
The cosine, WordNet and KL-divergence similarities with the question x_{q_i}	3
The cosine, WordNet and KL-divergence similarities with the questions other than x_{q_i}	3
The cosine, WordNet and KL-divergence similarities with previous and next sentences	6
Is this sentence x_j exactly x_{q_i} or one of the questions in $\{x_{q_1},\ldots,x_{q_m}\}$?	2
Is this sentence x_j in the three beginning sentences?	3
The relative position of this sentence x_j to questions	4
Is this sentence x_j share the same author with the question sentence x_{q_i}?	1
Is this sentence x_j in the same post with question sentences?	2
Is this sentence x_j in the same paragraph with question sentences?	2
The presence of greeting (e.g., "hi") and acknowledgement words in this sentence x_j	2
The length of this sentence x_j	1
The number of nouns, verbs and pronouns in this sentence x_j, respectively	3
$\Psi_h(\mathbf{x},\mathbf{y})$ (704 dimensions)	
For $\Psi_{hn}(\mathbf{x},\mathbf{y})$, the above 32 dimension features w.r.t. $4\times 4=16$ transition patterns	512
For $\Psi_{hc}(\mathbf{x},\mathbf{y})$, 12 types of pairwise or merged similarities w.r.t. 16 transition patterns	192
$\Psi_v(\mathbf{x},\mathbf{y})$ (32 dimensions)	
The transition patterns for any two non-contiguous labels in a label group	16
The transition patterns for any two contiguous labels in a label group	16

Table 1: Feature descriptions and demisions

$$\Psi_{hc}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m} \underbrace{\sum_{j\in\mathcal{C}}\sum_{k\in\mathcal{A}}}_{\text{Complete Edges}} \psi_{hc}(x_j,x_k,y_{ij},y_{ik}),$$

$$\begin{aligned}&\psi_{hn}(x_j,x_{j+1},y_{ij},y_{i,j+1})\\ &\quad= \Lambda(y_{ij},y_{i,j+1})\otimes\phi_{hn}(x_j,x_{j+1},y_{ij},y_{i,j+1}),\end{aligned}$$

$$\begin{aligned}&\psi_{hc}(x_j,x_k,y_{ij},y_{ik})\\ &\quad= \Lambda(y_{ij},y_{ik})\otimes\psi_{hc}(x_j,x_k,y_{ij},y_{ik})\end{aligned}$$

where $\Lambda(y_{ij},y_{ik})$ is a 16-dimensional vector. It indicates all 4×4 pairwise transition patterns of four types of labels, the context, answer, question and plain. Note that apart from previous work (Ding et al., 2008) we use *complete skip-chain (context-answer) edges* in $\Psi_{hc}(\mathbf{x},\mathbf{y})$.

The label group feature mapping $\Psi_v(\mathbf{x},\mathbf{y})$ is defined as follows,

$$\Psi_v(\mathbf{x},\mathbf{y}) = \sum_{j=1}^{n}\psi_v(x_j,\mathbf{y}_{.j}),$$

where $\psi_v(x_j,\mathbf{y}_{.j})$ encodes each label group pattern into a vector.

The detail descriptions and vector dimensions of the used features are listed in Table 1.

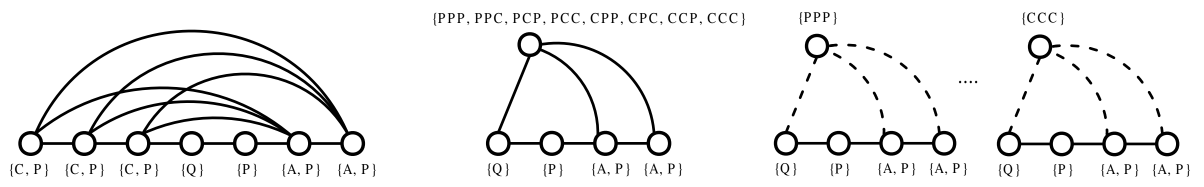
4 Structural SVMs and Inference

Given a training set $S=\{(\mathbf{x}^{(i)},\mathbf{y}^{(i)})\in\mathcal{X}\times\mathcal{Y}: i=1,\ldots,N\}$, we use the structural SVMs (Taskar et al., 2003; Tsochantaridis et al., 2005; Joachims et al., 2009) formulation, as shown in Optimization Problem 1 (OP1), to learn a weight vector $\mathbf{w}$.

OP 1 *(1-Slack Structural SVM)*

$$\begin{aligned}\min_{\mathbf{w},\xi\geq 0}\quad & \frac{1}{2}||\mathbf{w}||^2+\frac{C}{N}\xi\\ \textit{s.t.}\quad & \forall(\bar{\mathbf{y}}^{(1)},\ldots,\bar{\mathbf{y}}^{(N)})\in\mathcal{Y}^n,\\ & \frac{1}{N}\mathbf{w}^T\sum_{i=1}^{N}[\Psi(\mathbf{x}^{(i)},\mathbf{y}^{(i)})-\Psi(\mathbf{x}^{(i)},\bar{\mathbf{y}}^{(i)})]\\ & \geq\frac{1}{N}\sum_{i=1}^{N}\Delta(\mathbf{y}^{(i)},\bar{\mathbf{y}}^{(i)})-\xi,\end{aligned}$$

where ξ is a slack variable, $\Psi(\mathbf{x},\mathbf{y})$ is the joint feature mapping and $\Delta(\mathbf{y},\bar{\mathbf{y}})$ is the loss function that measures the loss caused by the difference between $\mathbf{y}$ and $\bar{\mathbf{y}}$. Though OP1 is already a quadratic optimization problem, directly using off-the-shelf quadratic optimization solver will fail, due to the large number of constraints. Instead, a cutting plane algorithm is used to efficiently solve this problem. For the details of the



(a) Original graph (b) Transformed graph (c) Decomposed graph

Figure 3: The equivalent transform of graphs

Algorithm 1 Exact Inference Algorithm

```
1: Input: $(\mathcal{C}_i, \mathcal{A}_i)$ for each $q_i$, $\mathbf{w}, \mathbf{x}, \mathbf{y}$
2: for $i \in \{1, \ldots, m\}$ do
3:     for $\mathcal{C}_s \subseteq \mathcal{C}_i$ do
4:         $[R(\mathcal{C}_s), \bar{\mathbf{y}}_{i\cdot}(\mathcal{C}_s)] \leftarrow \text{Viterbi}(\mathbf{w}, \mathbf{x}; \mathcal{C}_s)$
5:     end for
6:     $\mathcal{C}_s^* = \arg\max_{\mathcal{C}_s \subseteq \mathcal{C}_i} R(\mathcal{C}_s)$
7:     $\bar{\mathbf{y}}_{i\cdot}^* = \bar{\mathbf{y}}_{i\cdot}(\mathcal{C}_s^*)$
8: end for
9: return $\bar{\mathbf{y}}^*$
```

structural SVMs, please refer to (Tsochantaridis et al., 2005; Joachims et al., 2009).

The most essential and time-consuming step in structural SVMs is *finding the most violated constraint*, which is equivalent to solve

$$\arg\max_{\mathbf{y}\in\mathcal{Y}} \mathbf{w}^T \Psi(\mathbf{x}^{(i)}, \mathbf{y}) + \Delta(\mathbf{y}^{(i)}, \mathbf{y}). \qquad (2)$$

Without the ability to efficiently find the most violated constraint, the cutting plane algorithm is not tractable.

In the next sub-sections, we introduce the algorithms for finding the most violated constraint, also called loss-augmented inference. The algorithms are essential for the success of customizing structural SVMs to our problem.

4.1 Exact Inference

The exact inference algorithm is designed for a simplified model with two sub-mappings Ψ_n and Ψ_h, except Ψ_v.

One naive approach to finding the most violated constraint for the simplified model is to enumerate all the $2^{|\mathcal{C}|+|\mathcal{A}|}$ cases for each row of the label matrix. However, it would be intractable for large candidate sets.

An important property is that the context candidate set is usually much smaller than the whole number of sentences in a thread. This property enables us to design efficient and exact inference algorithm by transforming from the original graph representation in Figure 2 to the graphs in Figure 3. This graph transform merges all the nodes in the context candidate set $\mathcal{C}$ to one node with $2^{|\mathcal{C}|}$ possible labels.

We design an exact inference algorithm in Algorithm 1 based on the graph in Figure 3(c). The algorithm can be summarized in three steps: (1) enumerate all the $2^{|\mathcal{C}|}$ possible labels[1] for the merged node (line 3). (2) For each given label of the merged node, perform the Viterbi algorithm (Rabiner, 1989) on the decomposed graph (line 4) and store the Viterbi algorithm outputs in R and $\hat{\mathbf{y}}_{i\cdot}$. (3) From the $2^{|\mathcal{C}|}$ Viterbi algorithm outputs, select the one with highest score as the output (lines 6 and 7).

The use of the Viterbi algorithm is assured by the fact that there exists certain equivalence between the decomposed graph (Figure 3(c)) and a linear chain. By fixing the the label of the merged node, we could remove the dashed edges in the decomposed graph and regard the rest graph as a linear chain, which results in the Viterbi decoding.

4.2 Approximate Inference

The exact inference cannot handle the complete model with three sub-mappings, Ψ_n, Ψ_h, and Ψ_v, since the label group defeats the graph transform in Figure 3. Thus, we design two approximate algorithms by employing *undergenerating* and *overgenerating* approaches (Finley and Joachims, 2008).

First, we develop an *undergenerating* local greedy search algorithm shown in Algorithm 2. In the algorithm, there are two loops, inner and outer loops. The outer loop terminates when no labels change (steps 3-11). The inner loop enumerates the whole label matrix and greedily determines each label (step 7) by maximizing the Equation (2). Since the whole algorithm terminates only if

[1]Since the merged node is from context candidate set $\mathcal{C}$, enumerating its label is equivalent to enumerating subsets $\mathcal{C}_s$ of the candidate set $\mathcal{C}$

Algorithm 2 Greedy Inference Algorithm

```
1: Input: $\mathbf{w}, \mathbf{x}, \mathbf{y}$
2: initialize solution: $\bar{\mathbf{y}} \leftarrow \mathbf{y}_0$
3: repeat
4:     $\mathbf{y}' \leftarrow \bar{\mathbf{y}}$
5:     for $i \in \{1, \ldots, m\}$ do
6:         for $j \in \{1, \ldots, n\}$ do
7:             $\bar{y}^*_{ij} \leftarrow \arg\max_{\bar{y}_{ij}} \mathbf{w}^T \Psi(\mathbf{x}, \bar{\mathbf{y}}) + \triangle(\mathbf{y}, \bar{\mathbf{y}})$
8:             $\bar{y}_{ij} \leftarrow \bar{y}^*_{ij}$
9:         end for
10:    end for
11: until $\bar{\mathbf{y}} = \mathbf{y}'$
12: $\bar{\mathbf{y}}^* \leftarrow \bar{\mathbf{y}}$
13: return $\bar{\mathbf{y}}^*$
```

the label matrix does not change during the last outer loop. This indicates that at least a local optimal solution is obtained.

Second, an *overgenerating* method can be designed by using linear programming relaxation (Finley and Joachims, 2008). To save the space, we skip the details of this algorithm here.

5 Loss Functions

Structural SVMs allow users to customize the loss function $\triangle : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathcal{R}$ according to different system requirements. In this section, we introduce the loss functions used in our work.

Basic loss function. The simplest way to quantify the prediction quality is counting the number of wrongly predicted labels. Formally,

$$\triangle_b(\mathbf{y}, \bar{\mathbf{y}}) = \sum_{i=1}^{m} \sum_{j=1}^{n} I[y_{ij} \neq \bar{y}_{ij}], \quad (3)$$

where $I[.]$ is an indicative function that equals to one if the condition holds and zero otherwise.

Recall-vs-precision loss function. In practice, we may place different emphasis on recall and precision according to application settings. We could include this preference into the model by defining the following loss function,

$$\begin{aligned}\triangle_p(\mathbf{y}, \bar{\mathbf{y}}) \;=\; & \sum_{i=1}^{m} \sum_{j=1}^{n} I[y_{ij} \neq P, \bar{y}_{ij} = P] \cdot c_r \\ & + I[y_{ij} = P, \bar{y}_{ij} \neq P] \cdot c_p. \quad (4)\end{aligned}$$

This function penalizes the wrong prediction decreasing recall and that decreasing precision with two weights c_r and c_p respectively. Specifically, we denote the loss function with $c_p/c_r = 2$ and that with $c_r/c_p = 2$ by $\triangle_p^p$ and $\triangle_p^r$, respectively.

Various types of loss function can be defined in a similar fashion. To save the space, we skip the definitions of other loss functions and only use the above two types of loss functions to show the flexibility of our approach.

Items in the data set	#items
Thread	515
Post	2,035
Sentence	8,500
question annotation	1,407
context annotation	1,962
answer annotation	4,652
plain annotation	18,198

Table 2: The data statistics

6 Experiments

6.1 Experimental Setup

Corpus. We made use of the same data set as introduced in (Cong et al., 2008; Ding et al., 2008). Specifically, the data set includes about 591 threads from the forum TripAdvisor[2]. Each sentence in the threads is tagged with the labels 'question', 'context', 'answer', or 'plain' by two annotators. We removed 76 threads that have no question sentences or more than 40 sentences and 6 questions. The remaining 515 forum threads form our data set.

Table 2 gives the statistics on the data set. On average, each thread contains 3.95 posts and 2.73 questions, and each question has 1.39 context sentences and 3.31 answer sentences. Note that the number of annotations is much larger than the number of sentences because one sentence can be annotated with multiple labels.

Experimental Details. In all the experiments, we made use of linear models for the sake of computational efficiency. As a preprocessing step, we normalized the value of each feature value into the interval $[0, 1]$ and then followed the heuristic used in SVM-light (Joachims, 1998) to set C to $1/||x||^2$, where $||x||$ is the average length of input samples (in our case, sentences). The tolerance parameter ϵ was set to 0.1 (the value also used in (Cai

[2]TripAdvisor (`http://www.tripadvisor.com/ForumHome`) is one of the most popular travel forums

and Hofmann, 2004)) in all the runs of the experiments.

Evaluation. We calculated the standard precision (P), recall (R) and F_1-score (F_1) for both tasks (context extraction and answer extraction). All the experimental results were obtained through 5-fold cross validation.

6.2 Baseline Methods

We employed binary SVMs (B-SVM), multiclass SVMs (M-SVM), and C4.5 (Quinlan, 1993) as our baseline methods:

B-SVM. We trained two binary SVMs for context extraction (context vs. non-context) and answer extraction (answer vs. non-answer), respectively. We used the feature mapping $\phi_{q_i}(x_j)$ defined in Equation (1) while training the binary SVM models.

M-SVM. We extended the binary SVMs by training multiclass SVMs for three category labels (*context*, *answer*, *plain*).

C4.5. This decision tree algorithm solved the same classification problem as binary SVMs and made use of the same set of features.

Method	Δ_b	P (%)	R (%)	F_1 (%)
Context Extraction				
C4.5	–	74.2	68.7	71.2
B-SVM	–	78.3	72.2	74.9
M-SVM	–	68.0	77.6	72.1
S-SVM	8.86	75.6	71.7	73.4
S-SVM-H	8.60	77.5	75.5	76.3
S-SVM-HC*	8.65	77.9	74.1	75.8
S-SVM-HC	8.62	77.5	75.2	76.2
S-SVM-HCV*	8.08	79.5	79.6	79.5
S-SVM-HCV	**7.98**	79.7	80.2	**79.9**
Answer Extraction				
C4.5	–	61.3	45.2	51.8
B-SVM	–	69.7	42.0	51.8
M-SVM	–	63.2	51.5	55.8
S-SVM	8.86	67.0	48.0	55.6
S-SVM-H	8.60	66.9	49.7	56.7
S-SVM-HC*	8.65	66.5	49.4	56.4
S-SVM-HC	8.62	65.7	51.5	57.4
S-SVM-HCV*	8.08	65.5	58.7	61.7
S-SVM-HCV	**7.98**	65.1	61.2	**63.0**

Table 3: The effectiveness of our approach

6.3 Modeling Sentence Relations and Question Interactions

We demonstrate in Table 3 that our approach can make use of the three types of relation among sentences well to boost the performance.

In Table 3, S-SVM represents the structural SVMs only using the node features $\Psi_n(\mathbf{x}, \mathbf{y})$. The suffixes *H*, *C*, and *V* denote the models using *horizontal sequential edges*, *complete skip-chain edges* and *vertical label groups*, respectively. The suffixes *C** and *V** denote the models using *incomplete skip-chain edges* and *vertical sequential edges* proposed in (Ding et al., 2008), as shown in Figures 2(a) and 2(c). All the structural SVMs were trained using basic loss function Δ_b in Equation (3). From Table 3, we can observe the following advantages of our approaches.

Overall improvement. Our structural approach steadily improves the extraction as more types of relation (corresponding to more types of edge) are included. The best results obtained by using the three types of relation together improve the baseline methods binary SVMs by about 6% and 20% in terms of F_1 values for context extraction and answer extraction, respectively.

The usefulness of relations. The relations encoded by horizontal sequential edges and label groups are useful for both context extraction and answer extraction. The relation encoded by complete skip-chain edges is useful for answer extraction. The complete skip-chain edges not only avoid preprocessing but also boost the performance when compared with the preprocessed skip-chain edges. The label groups improve the vertical sequential edges.

Interactions among questions. The interactions encoded by label groups are especially useful. We conducted significance tests (sign test) on the experimental results. The test result shows that S-SVM-HCV outperforms all the other methods without vertical edges statistically significantly (p-value < 0.01). Our proposed graphical representation in Figure 2(d) eases us to model the complex interactions. In comparison, the 2D model in Figure 2(c) used in previous work (Ding et al., 2008) can only model the interaction between adjacent questions.

6.4 Loss Function Results

We report in Table 4 the comparison between structural SVMs using different loss functions. Note that Δ_p^p prefers precision and Δ_p^r prefers recall. From Table 4, we can observe that the experimental results also exhibit this kind of system

Method	P (%)	R (%)	F_1 (%)
Context Extraction			
S-SVM-HCV-$\triangle_b$	79.7	80.2	79.9
S-SVM-HCV-$\triangle_p^p$	**82.0**	70.3	75.6
S-SVM-HCV-$\triangle_p^r$	75.7	**84.2**	79.7
Answer Extraction			
S-SVM-HCV-$\triangle_b$	65.1	61.2	63.0
S-SVM-HCV-$\triangle_p^p$	**71.8**	52.2	60.2
S-SVM-HCV-$\triangle_p^r$	61.8	**66.1**	63.7

Table 4: The use of different loss functions

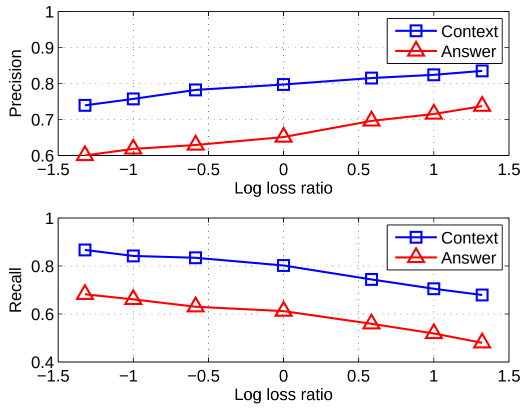


Figure 4: Balancing between precision and recall

preference. Moreover, we further demonstrate the capability of the loss function Δ_p in Figure 4. The curves are achieved by varying the ratio between two parameters c_p/c_r in Equation (4). The curves confirm our intuition: when $\log(c_p/c_r)$ becomes larger, the precisions increase but the recalls decrease and vice versa.

7 Related work

Previous work on extracting questions, answers and contexts is most related with our work. Cong et al. (2008) proposed a supervised approach for question detection and an unsupervised approach for answer detection without considering contexts. Ding et al. (2008) used CRFs to detect contexts and answers of questions from forum threads.

Some researches on summarizing discussion threads and emails are related to our work, too. Zhou and Hovy (2005) segmented internet relay chat, clustered segments into sub-topics, and identified responding segments of the first segment in each sub-topic by assuming the first segment to be focus. In (Nenkova and Bagga, 2003; Wan and McKeown, 2004; Rambow et al., 2004), email summaries were organized by extracting overview sentences as discussion issues. The work (Shrestha and McKeown, 2004) used RIPPER as a classifier to detect interrogative questions and their answers then used the resulting question and answer pairs as summaries. We also note the existing work on extracting knowledge from discussion threads. Huang et al. (2007) used SVMs to extract input-reply pairs from forums for chatbot knowledge. Feng et al. (2006) implemented a discussion-bot which used cosine similarity to match students' query with reply posts from an annotated corpus of archived threaded discussions.

Moreover, extensive researches have been done within the area of question answering (Burger et al., 2006; Jeon et al., 2005; Harabagiu and Hickl, 2006; Cui et al., 2005; Dang et al., 2006). They mainly focused on using sophisticated linguistic analysis to construct answer from a large document collection.

8 Conclusion and Future Work

We have proposed a new form of graphical representation for modeling the problem of extracting contexts and answers of questions from online forums and then customized structural SVM approach to solve it.

The proposed graphical representation is able to naturally express three types of relation among sentences: relation between successive sentences, relation between context sentences and answer sentences, and relation between multiple labels for one sentence. The representation also enables us to address interactions among questions. We also developed the inference algorithms for the structural SVM model by exploiting the special structure of thread discussions.

Experimental results on a real data set show that our approach significantly improves the baseline methods by effectively utilizing various types of relation among sentences.

Our future work includes: (a) to summarize threads and represent the forum threads in question-context-answer triple, which will change the organization of online forums; and (b) to enhance QA services (e.g., Yahoo! Answers) by the contents extracted from online forums.

Acknowledgement

The authors would like to thank the anonymous reviewers for their comments to improve this paper.

References

John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, Chin-Yew Lin, Steve Maiorano, George Miller, Dan Moldovan, Bill Ogden, John Prager, Ellen Riloff, Amit Singhal, Rohini Shrihari, Tomek Strzalkowski, Ellen Voorhees, and Ralph Weishedel. 2006. Issues, tasks and program structures to roadmap research in question and answering (qna). *ARAD: Advanced Research and Development Activity (US)*.

Lijuan Cai and Thomas Hofmann. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of CIKM*, pages 78–87.

Gao Cong, Long Wang, Chin-Yew Lin, and Young-In Song. 2008. Finding question-answer pairs from online forums. In *Proceedings of SIGIR*, pages 467–474.

Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of SIGIR*, pages 400–407.

Hoa Dang, Jimmy Lin, and Diane Kelly. 2006. Overview of the trec 2006 question answering track. In *Proceedings of TREC*, pages 99–116.

Shilin Ding, Gao Cong, Chin-Yew Lin, and Xiaoyan Zhu. 2008. Using conditional random field to extract contexts and answers of questions from online forums. In *Proceedings of ACL*, pages 710–718.

Donghui Feng, Erin Shaw, Jihie Kim, and Eduard H. Hovy. 2006. An intelligent discussion-bot for answering student queries in threaded discussions. In *Proceedings of IUI*, pages 171–177.

Thomas Finley and Thorsten Joachims. 2008. Training structural SVMs when exact inference is intractable. In *Proceedings of ICML*, pages 304–311.

Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 364–372.

Sanda M. Harabagiu and Andrew Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of ACL*, pages 905–912.

Jizhou Huang, Ming Zhou, and Dan Yang. 2007. Extracting chatbot knowledge from online discussion forums. In *Proceedings of IJCAI*, pages 423–428.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of CIKM*, pages 84–90.

Thorsten Joachims, Thomas Finley, and Chun-Nam Yu. 2009. Cutting-plane training of structural SVMs. *Machine Learning*.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of ECML*, pages 137–142.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289.

Ani Nenkova and Amit Bagga. 2003. Facilitating email thread access by extractive summary generation. In *Proceedings of RANLP*, pages 287–296.

Nam Nguyen and Yunsong Guo. 2007. Comparisons of sequence labeling algorithms and extensions. In *Proceedings of ICML*, pages 681–688.

John Quinlan. 1993. *C4.5: programs for machine learning*. Morgan Kaufmann Publisher Incorporation.

Lawrence Rabiner. 1989. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of IEEE*, pages 257–286.

Owen Rambow, Lokesh Shrestha, John Chen, and Chirsty Lauridsen. 2004. Summarizing email threads. In *Proceedings of HLT-NAACL*, pages 105–108.

Lokesh Shrestha and Kathleen McKeown. 2004. Detection of question-answer pairs in email conversations. In *Proceedings of COLING*, pages 889–895.

Charles Sutton and Andrew McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. Technical Report 04-49.

Benjamin Taskar, Carlos Guestrin, and Daphne Koller. 2003. Max-margin markov networks. In *Advances in Neural Information Processing Systems 16*. MIT Press.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484.

Stephen Wan and Kathy McKeown. 2004. Generating overview summaries of ongoing email thread discussions. In *Proceedings of COLING*, pages 549–555.

Liang Zhou and Eduard Hovy. 2005. Digesting virtual ”geek” culture: The summarization of technical internet relay chats. In *Proceedings of ACL*, pages 298–305.

Jun Zhu, Zaiqing Nie, Ji-Rong Wen, Bo Zhang, and Wei-Ying Ma. 2005. 2d conditional random fields for web information extraction. In *Proceedings of ICML*, pages 1044–1051.