Joint Multitask Learning for Community Question Answering Using Task-Specific Embeddings

Shafiq Joty†, Lluís Màrquez* and Preslav Nakov‡
†Nanyang Technological University, Singapore
*Amazon, Barcelona, Spain
‡Qatar Computing Research Institute, HBKU, Qatar
srjoty@ntu.edu.sg, lluismv@amazon.com, pnakov@qf.org.qa

Abstract

We address jointly two important tasks for Question Answering in community forums: given a new question, (*i*) *find related existing questions*, and (*ii*) *find relevant answers to this new question*. We further use an auxiliary task to complement the previous two, i.e., (*iii*) *find good answers with respect to the thread question in a question-comment thread*. We use deep neural networks (DNNs) to learn meaningful task-specific embeddings, which we then incorporate into a conditional random field (CRF) model for the multitask setting, performing joint learning over a complex graph structure. While DNNs alone achieve competitive results when trained to produce the embeddings, the CRF, which makes use of the embeddings and the dependencies between the tasks, improves the results significantly and consistently across a variety of evaluation metrics, thus showing the complementarity of DNNs and structured learning.

1 Introduction and Motivation

Question answering web forums such as Stack-Overflow, Quora, and Yahoo! Answers usually organize their content in topically-defined forums containing multiple *question–comment threads*, where a question posed by a user is often followed by a possibly very long list of comments by other users, supposedly intended to answer the question. Many forums are not moderated, which often results in noisy and redundant content.

Within community Question Answering (cQA) forums, two subtasks are of special relevance when a user poses a new question to the website (Hoogeveen et al., 2018; Lai et al., 2018): (*i*) finding similar questions (*question-question relatedness*), and (*ii*) finding relevant answers to the new question, if they already exist (*answer selection*).

---

*Work conducted while this author was at QCRI, HBKU.

Both subtasks have been the focus of recent research as they result in end-user applications. The former is interesting for a user who wants to explore the space of similar questions in the forum and to decide whether to post a new question. It can also be relevant for the forum owners as it can help detect redundancy, eliminate question duplicates, and improve the overall forum structure. Subtask (*ii*) on the other hand is useful for a user who just wants a quick answer to a specific question, without the need of digging through the long answer threads and winnowing good from bad comments or without having to post a question and then wait for an answer.

Obviously, the two subtasks are interrelated as the information needed to answer a new question is usually found in the threads of highly related questions. Here, we focus on jointly solving the two subtasks with the help of yet another related subtask, i.e., determining whether a comment within a question-comment thread is a good answer to the question heading that thread.

An example is shown in Figure 1. A new question q is posed for which several potentially related questions are identified in the forum (e.g., by using an information retrieval system); q_i in the example is one of these existing questions. Each retrieved question comes with an associated thread of comments; c_m^i represents one comment from the thread of question q_i. Here, c_m^i is a *good* answer for q_i, q_i is indeed a question *related* to q, and consequently c_m^i is a *relevant* answer for the new question q. This is the setting of SemEval-2016 Task 3, and we use its benchmark datasets.

Our approach has two steps. First, a deep neural network (DNN) in the form of a feed-forward neural network is trained to solve each of the three subtasks separately, and the subtask-specific hidden layer activations are taken as embedded feature representations to be used in the second step.

q: **"How can I extend a family visit visa?"**

q_i: "Dear All; I wonder if anyone knows the procedure how I can extend the family visit visa for my wife beyond 6 months. I already extended it for 5 months and is 6 months running. I would like to get it extended for couple of months more. Any suggestion is highly appreciable. Thanks"

c_m^i: "You can get just another month's extension before she completes 6 months by presenting to immigration office a confirmed booking of her return ticket which must not exceed 7 months."

Figure 1: Example of the three pieces of information in the cQA problems addressed in this paper.

Then, a conditional random field (CRF) model uses these embeddings and performs joint learning with global inference to exploit the dependencies between the subtasks.

A key strength of DNNs is their ability to learn nonlinear interactions between underlying features through specifically-designed hidden layers, and also to learn the features (e.g., vectors for words and documents) automatically. This capability has led to gains in many unstructured output problems. DNNs are also powerful for structured output problems. Previous work has mostly relied on recurrent or recursive architectures to propagate information through hidden layers, but has been disregarding the modeling strength of structured conditional models, which use global inference to model consistency in the output structure (i.e., the class labels of all nodes in a graph). In this work, we explore the idea that combining simple DNNs with structured conditional models can be an effective and efficient approach for cQA subtasks that offers the best of both worlds.

Our experimental results show that: (*i*) DNNs already perform very well on the question-question similarity and answer selection subtasks; (*ii*) strong dependencies exist between the subtasks under study, especially answer-goodness and question-question-relatedness influence answer-selection significantly; (*iii*) the CRFs exploit the dependencies between subtasks, providing sizeably better results that are on par or above the state of the art. In summary, we demonstrate the effectiveness of this marriage of DNNs and structured conditional models for cQA subtasks, where a feed-forward DNN is first used to build vectors for each individual subtask, which are then "reconciled" in a multitask CRF.

2 Related Work

Various neural models have been applied to cQA tasks such as *question-question similarity* (dos Santos et al., 2015; Lei et al., 2016; Wang et al., 2018) and *answer selection* (Wang and Nyberg, 2015; Qiu and Huang, 2015; Tan et al., 2015; Chen and Bunescu, 2017; Wu et al., 2018). Most of this work used advanced neural network architectures based on convolutional neural networks (CNN), long short-term memory (LSTM) units, attention mechanism, etc. For instance, dos Santos et al. (2015) combined CNN and bag of words for comparing questions. Tan et al. (2015) adopted an attention mechanism over bidirectional LSTMs to generate better answer representations, and Lei et al. (2016) combined recurrent and CNN models for question representation. In contrast, here we use a simple DNN model, i.e., a feed-forward neural network, which we only use to generate task-specific embeddings, and we defer the joint learning with global inference to the structured model.

From the perspective of modeling cQA subtasks as structured learning problems, there is a lot of research trying to exploit the correlations between the comments in a question–comment thread. This has been done from a feature engineering perspective, by modeling a comment in the context of the entire thread (Barrón-Cedeño et al., 2015), but more interestingly by considering a thread as a structured object, where comments are to be classified as *good* or *bad* answers collectively. For example, Zhou et al. (2015) treated the answer selection task as a sequence labeling problem and used recurrent convolutional neural networks and LSTMs. Joty et al. (2015) modeled the relations between pairs of comments at any distance in the thread, and combined the predictions of local classifiers using graph-cut and Integer Linear Programming. In a follow up work, Joty et al. (2016) also modeled the relations between all pairs of comments in a thread, but using a fully-connected pairwise CRF model, which is a joint model that integrates inference within the learning process using global normalization. Unlike these models, we use DNNs to induce task-specific embeddings, and, more importantly, we perform multitask learning of three different cQA subtasks, thus enriching the relational structure of the graphical model.

We solve the three cQA subtasks jointly, in a multitask learning framework. We do this using the

datasets from the SemEval-2016 Task 3 on *Community Question Answering* (Nakov et al., 2016b), which are annotated for the three subtasks, and we compare against the systems that participated in that competition. In fact, most of these systems did not try to exploit the interaction between the subtasks or did so only as a pipeline. For example, the top two systems, SUPER TEAM (Mihaylova et al., 2016) and KELP (Filice et al., 2016), stacked the predicted labels from two subtasks in order to solve the main answer selection subtask using SVMs. In contrast, our approach is neural, it is based on joint learning and task-specific embeddings, and it is also lighter in terms of features.

In work following the competition, Nakov et al. (2016a) used a triangulation approach to answer ranking in cQA, modeling the three types of similarities occurring in the triangle formed by the original question, the related question, and an answer to the related comment. However, theirs is a pairwise ranking model, while we have a joint model. Moreover, they focus on one task only, while we use multitask learning. Bonadiman et al. (2017) proposed a multitask neural architecture where the three tasks are trained together with the same representation. However, they do not model comment-comment interactions in the same question-comment thread nor do they train task-specific embeddings, as we do.

The general idea of combining DNNs and structured models has been explored recently for other NLP tasks. Collobert et al. (2011) used Viterbi inference to train their DNN models to capture dependencies between word-level tags for a number of sequence labeling tasks: part-of-speech tagging, chunking, named entity recognition, and semantic role labeling. Huang et al. (2015) proposed an LSTM-CRF framework for such tasks. Ma and Hovy (2016) included a CNN in the framework to compute word representations from character-level embeddings. While these studies consider tasks related to constituents in a sentence, e.g., words and phrases, we focus on methods to represent comments and to model dependencies between comment-level tags. We also experiment with arbitrary graph structures in our CRF model to model dependencies at different levels.

3 Learning Approach

Let q be a newly-posed question, and c_m^i denote the m-th comment ($m \in \{1, 2, \ldots, M\}$) in the

answer thread for the i-th potentially related question q_i ($i \in \{1, 2, \ldots, I\}$) retrieved from the forum. We can define three cQA subtasks: (*A*) classify each comment c_m^i in the thread for question q_i as *Good* vs. *Bad* with respect to q_i; (*B*) determine, for each retrieved question q_i, whether it is *Related* to the new question q in the sense that a good answer to q_i might also be a good answer to q; and finally, (*C*) classify each comment c_m^i in each answer thread as either *Relevant* or *Irrelevant* with respect to the new question q.

Let $y_{i,m}^a \in \{Good, Bad\}$, $y_i^b \in \{Related, Not\text{-}related\}$, and $y_{i,m}^c \in \{Relevant, Irrelevant\}$ denote the corresponding output labels for subtasks A, B, and C, respectively. As argued before, subtask C depends on the other two subtasks. Intuitively, if c_m^i is a good comment with respect to the existing question q_i, and q_i is related to the new question q (subtask A), then c_m^i is likely to be a relevant answer to q. Similarly, subtask B can benefit from subtask C: if comment c_m^i in the answer thread of q_i is relevant with respect to q, then q_i is likely to be related to q.

We propose to exploit these inherent correlations between the cQA subtasks as follows: (*i*) by modeling their interactions in the input representations, i.e., in the feature space of (q, q_i, c_m^i), and more importantly, (*ii*) by capturing the dependencies between the output variables $(y_{i,m}^a, y_i^b, y_{i,m}^c)$. Moreover, we cast each cQA subtask as a structured prediction problem in order to model the dependencies between output variables of the same type. Our intuition is that if two comments c_m^i and c_n^i in the same thread are similar, then they are likely to have the same labels for both subtask A and subtask C, i.e., $y_{i,m}^a \approx y_{i,n}^a$, and $y_{i,m}^c \approx y_{i,n}^c$. Similarly, if two pre-existing questions q_i and q_j are similar, they are also likely to have the same labels, i.e., $y_i^b \approx y_j^b$.

Our framework works in two steps. First, we use a DNN, specifically, a feed-forward NN, to learn task-specific embeddings for the three subtasks, i.e., output embeddings $\mathbf{x}_{i,m}^a$, $\mathbf{x}_i^b$ and $\mathbf{x}_{i,m}^c$ for subtasks A, B and C (Figure 2a). The DNN uses syntactic and semantic embeddings of the input elements, their interactions, and other similarity features between them and, as a by-product, learns the output embeddings for each subtask.

In the second step, a structured conditional model operates on subtask-specific embeddings from the DNNs and captures the dependencies between the

Input embedding	Interaction layer	Task-specific embedding

(a) Our feed-forward neural networks

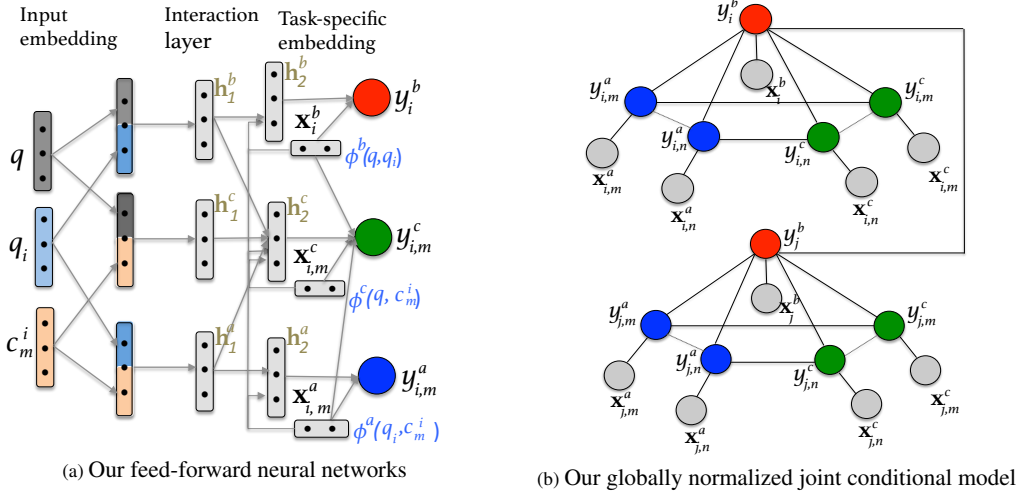(b) Our globally normalized joint conditional model

Figure 2: Graphical representation of our cQA framework. On the left (a), we have three feed-forward neural networks to learn task-specific embeddings for the three cQA subtasks. On the right (b), a global conditional random field (CRF) models intra- and inter-subtask dependencies.

subtasks, between existing questions, and between comments for an existing question (Figure 2b). Below, we describe the two steps in detail.

3.1 Neural Models for cQA Subtasks

Figure 2a depicts our complete neural framework for the three subtasks. The input is a tuple (q, q_i, c_m^i) consisting of a new question q, a retrieved question q_i, and a comment c_m^i from q_i's answer thread. We first map the input elements to fixed-length vectors $(\mathbf{z}_q, \mathbf{z}_{q_i}, \mathbf{z}_{c_m^i})$ using their syntactic and semantic *embeddings*. Depending on the requirements of the subtasks, the network then models the interactions between the inputs by passing their embeddings through non-linear hidden layers $\nu(\cdot)$. Additionally, the network also considers *pairwise* similarity features $\phi(\cdot)$ between two input elements that go directly to the output layer, and also through the last hidden layer. The pairwise features together with the activations at the final hidden layer constitute the task-specific embeddings for each subtask t: $\mathbf{x}_i^t = [\nu^t(\cdot), \phi^t(\cdot)]$. The final layer defines a Bernoulli distribution for each subtask $t \in \{a, b, c\}$:

$$p(y_i^t | q, q_i, c_m^i, \theta) = \text{Ber}(y_i^t | \text{sig}(\mathbf{w}_t^T \mathbf{x}_i^t)) \quad (1)$$

where $\mathbf{x}_i^t$, $\mathbf{w}_t$, and y_i^t are the task-specific embedding, the output layer weights, and the prediction variable for subtask t, respectively, and $\text{sig}(\cdot)$ refers to the sigmoid function.

We train the models by minimizing the cross-entropy between the predicted distribution and the gold labels. The main difference between the models is how they compute the task-specific embeddings $\mathbf{x}_i^t$ for subtask t.

Neural Model for Subtask A. The feed-forward network for subtask A is shown in the lower part of Figure 2a. To determine whether a comment c_m^i is *good* with respect to the thread question q_i, we model the interactions between c_m^i and q_i by merging their embeddings $\mathbf{z}_{c_m^i}$ and $\mathbf{z}_{q_i}$, and passing them through a hidden layer:

$$\mathbf{h}_1^a = f(U^a [\mathbf{z}_{q_i}, \mathbf{z}_{c_m^i}]) \quad (2)$$

where U^a is the weight matrix from the inputs to the first hidden units, f is a non-linear activation function. The activations are then fed to a final subtask-specific hidden layer, which combines these signals with the pairwise similarity features $\phi^a(q_i, c_m^i)$. Formally,

$$\mathbf{h}_2^a = f(V^a [\mathbf{h}_1^a, \phi^a(q_i, c_m^i)]) \quad (3)$$

where V^a is the weight matrix. The task-specific output embedding is formed by merging $\mathbf{h}_2^a$ and $\phi^a(q_i, c_m^i)$; $\mathbf{x}_{i,m}^a = [\mathbf{h}_2^a, \phi^a(q_i, c_m^i)]$.

Neural Model for Subtask B. To determine whether an existing question q_i is *related* to the new question q, we model the interactions between q and q_i using their embeddings and pairwise similarity features similarly to subtask A.

The upper part of Figure 2a shows the network. The transformation is defined as follows:

$$\mathbf{h}_1^b = f(U^b [\mathbf{z}_q, \mathbf{z}_{q_i}]); \mathbf{h}_2^b = f(V^b [\mathbf{h}_1^b, \phi^b(q, q_i)])$$

4199

where U^b and V^b are the weight matrices in the first and second hidden layer. The task-specific embedding is formed by $\mathbf{x}_i^b = [\mathbf{h}_2^b, \phi^b(q, q_i)]$.

Neural Model for Subtask C. The network for subtask C is shown in the middle of Figure 2a. To decide if a comment c_m^i in the thread of q_i is *relevant* to q, we consider how related q_i is to q, and how useful c_m^i is to answer q_i. Again, we model the direct interactions between q and c_m^i using pairwise features $\phi^c(q, c_m^i)$ and a hidden layer transformation $\mathbf{h}_1^c = f(U^c[\mathbf{z}_q, \mathbf{z}_{c_m^i}])$, where U^c is a weight matrix. We then include a second hidden layer to combine the activations from different inputs and pairwise similarity features. Formally,

$$\mathbf{h}_2^c = f(V^c[\mathbf{h}_1^a, \mathbf{h}_1^b, \mathbf{h}_1^c, \phi^a(q_i, c_m^i), \phi^b(q, q_i), \phi^c(q, c_m^i)])$$

The final task-specific embedding for subtask C is formed as $\mathbf{x}_i^c = [\mathbf{h}_2^c, \phi^a(q_i, c_m^i), \phi^b(q, q_i), \phi^c(q, c_m^i)]$.

3.2 Joint Learning with Global Inference

One simple way to exploit the interdependencies between the subtask-specific embeddings ($\mathbf{x}_{i,m}^a$, $\mathbf{x}_i^b$, $\mathbf{x}_{i,m}^c$) is to precompute the predictions for some subtasks (A and B), and then to use the predictions as features for the other subtask (C). However, as shown later in Section 6, such a *pipeline* approach propagates errors from one subtask to the subsequent ones. A more robust way is to build a joint model for all subtasks.

We could use the full DNN network in Figure 2a to learn the classification functions for the three subtasks *jointly* as follows:

$$p(y_{i,m}^a, y_i^b, y_{i,m}^c | \theta) = p(y_{i,m}^a | \theta_a) p(y_i^b | \theta_b) p(y_{i,m}^c | \theta_c) \quad (4)$$

where $\theta = [\theta_a, \theta_b, \theta_c]$ are the model parameters.

However, this has two key limitations: (*i*) it assumes conditional independence between the subtasks given the parameters; (*ii*) the scores are normalized locally, which leads to the so-called *label bias* problem (Lafferty et al., 2001), i.e., the features for one subtask would have no influence on the other subtasks.

Thus, we model the dependencies between the output variables by learning (globally normalized) node and edge factor functions that jointly optimize a global performance criterion. In particular, we represent the cQA setting as a large undirected graph $G=(V, E)=(V_a \cup V_b \cup V_c, E_{aa} \cup E_{bb} \cup E_{cc} \cup E_{ac} \cup E_{bc} \cup E_{ab})$. As shown in Figure 2b, the graph contains six subgraphs:

$G_a=(V_a, E_{aa})$, $G_b=(V_b, E_{bb})$ and $G_c=(V_c, E_{cc})$ are associated with the three subtasks, while the bipartite subgraphs $G_{ac}=(V_a \cup V_c, E_{ac})$, $G_{bc}=(V_b \cup V_c, E_{bc})$ and $G_{ab}=(V_a \cup V_b, E_{ab})$ connect nodes across tasks.

We associate each node $u \in V_t$ with an input vector $\mathbf{x}_u$, representing the embedding for subtask t, and an output variable y_u, representing the class label for subtask t. Similarly, each edge $(u, v) \in E_{st}$ is associated with an input feature vector $\mu(\mathbf{x}_u, \mathbf{x}_v)$, derived from the node-level features, and an output variable $y_{uv} \in \{1, 2, \cdots, L\}$, representing the state transitions for the pair of nodes.[1] For notational simplicity, here we do not distinguish between comment and question nodes, rather we use u and v as general indices. We define the following joint conditional distribution:

$$p(\mathbf{y}|\theta, \mathbf{x}) = \frac{1}{Z(\theta, \mathbf{x})} \prod_{t \in \tau} \Big[\prod_{u \in V_t} \psi_n(y_u | \mathbf{x}, \mathbf{w}_n^t) \Big]$$
$$\prod_{(s,t) \in \tau \times \tau} \Big[\prod_{(u,v) \in E_{st}} \psi_e(y_{uv} | \mathbf{x}, \mathbf{w}_e^{st}) \Big] \quad (5)$$

where $\tau = \{a, b, c\}$, $\psi_n(\cdot)$ and $\psi_e(\cdot)$ are node and edge *factors*, respectively, and $Z(\cdot)$ is a global normalization constant. We use log-linear factors:

$$\psi_n(y_u | \mathbf{x}, \mathbf{w}_n^t) = \exp(\sigma(y_u, \mathbf{x})^T \mathbf{w}_n^t) \quad (6)$$
$$\psi_e(y_{uv} | \mathbf{x}, \mathbf{w}_e^{st}) = \exp(\sigma(y_{uv}, \mathbf{x})^T \mathbf{w}_e^{st}) \quad (7)$$

where $\sigma(\cdot)$ 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 *label bias* problem, allowing them to take long-range interactions into account. The objective in Equation 5 is a convex function, and thus we can use gradient-based methods to find the global optimum. The gradients have the following form:

$$f'(\mathbf{w}_n^t) = \sum_{u \in V_t} \sigma(y_u, \mathbf{x}) - \mathbb{E}[\sigma(y_u, \mathbf{x})] \quad (8)$$
$$f'(\mathbf{w}_e^{st}) = \sum_{(u,v) \in E_{st}} \sigma(y_{uv}, \mathbf{x}) - \mathbb{E}[\sigma(y_{uv}, \mathbf{x})] \quad (9)$$

where $\mathbb{E}[\phi(\cdot)]$ is the expected feature vector.

Training and Inference. Traditionally, CRFs have been trained using offline methods like LBFGS (Murphy, 2012). Online training using

---

[1] To avoid visual clutter, the input features and the output variables for the edges are not shown in Figure 2b.

4200

first-order methods such as stochastic gradient descent was proposed by Vishwanathan et al. (2006). Since our DNNs are trained with the RMSprop online adaptive algorithm (Tieleman and Hinton, 2012), in order to compare our two models, we use RMSprop to train our CRFs as well.

For our CRF models, we use Belief Propagation, or BP, (Pearl, 1988) for inference. BP converges to an exact solution for trees. However, exact inference is intractable for graphs with loops. Despite this, Pearl (1988) advocated for the use of BP in loopy graphs as an approximation. Even though BP only gives approximate solutions, it often works well in practice for loopy graphs (Murphy et al., 1999), outperforming other methods such as mean field (Weiss, 2001).

Variations of Graph Structures. A crucial advantage of our CRFs is that we can use arbitrary graph structures, which allows us to capture dependencies between different types of variables: (*i*) *intra-subtask*, for variables of the same subtask, e.g., y_i^b and y_j^b in Figure 2b, and (*ii*) *across-subtask*, for variables of different subtasks.

For intra-subtask, we explore *null* (i.e., no connection between nodes) and *fully-connected* relations. For subtasks A and C, the intra-subtask connections are restricted to the nodes inside a thread, e.g., we do not connect $y_{i,m}^c$ and $y_{j,m}^c$ in Figure 2b.

For across-subtask, we explored three types of connections depending on the subtasks involved: (*i*) *null* or no connection between subtasks, (*ii*) *1:1* connection for A-C, where the corresponding nodes of the two subtasks in a thread are connected, e.g., $y_{i,m}^a$ and $y_{i,m}^c$ in Figure 2b, and (*iii*) *M:1* connection to B, where we connect all the nodes of C or A to the thread-level B node. Each configuration of intra- and across-connections yields a different CRF model. Figure 2b shows one such model for two threads each containing two comments, where all subtasks have fully-connected intra-subtask links, 1:1 connection for A-C, and M:1 for C-B and A-B.

4 Features for the DNN Models

We have two types of features: (*i*) *input embeddings*, for q, q_i and c_m^i, and (*ii*) *pairwise features*, for (q, q_i), (q, c_m^i), and (q_i, c_m^i) — see Figure 2a.

4.1 Input Embeddings

We use three types of pre-trained vectors to represent a question (q or q_i) or a comment (c_m^i):

GOOGLE VECTORS. 300-dimensional embedding vectors, trained on 100 billion words from Google News (Mikolov et al., 2013). The embedding for a question (or comment) is the average of the word embeddings it is composed of.

SYNTAX. We parse the question (or comment) using the Stanford neural parser (Socher et al., 2013), and we use the final 25-dimensional vector produced internally as a by-product of parsing.

QL VECTORS. We use fine-tuned word embeddings pretrained on all the available in-domain Qatar Living data (Mihaylov and Nakov, 2016).

4.2 Pairwise Features

We extract pairwise features for each of (q, q_i), (q, c_m^i), and (q_i, c_m^i) pairs. These include:

COSINES. We compute cosines using the above vectors: $\cos(q, q_i)$, $\cos(q, c_m^i)$ and $\cos(q_i, c_m^i)$.

MT FEATURES. We use the following machine translation evaluation metrics: (1) BLEU (Papineni et al., 2002); (2) NIST (Doddington, 2002); (3) TER v0.7.25 (Snover et al., 2006); (4) METEOR v1.4 (Lavie and Denkowski, 2009); (5) Unigram PRECISION; (6) Unigram RECALL.

BLEU COMPONENTS. We further use various components involved in the computation of BLEU:[2] n-gram precisions, n-gram matches, total number of n-grams (n=1,2,3,4), lengths of the hypotheses and of the reference, length ratio between them, and BLEU's brevity penalty.

QUESTION-COMMENT RATIO. (1) question-to-comment count ratio in terms of sentences/tokens/nouns/verbs/adjectives/adverbs/pronouns; (2) question-to-comment count ratio of words that are not in WORD2VEC's Google News vocabulary.

4.3 Node Features

COMMENT FEATURES. These include number of (1) nouns/verbs/adjectives/adverbs/pronouns, (2) URLs/images/emails/phone numbers, (3) tokens/sentences, (4) positive/negative smileys, (5) single/double/triple exclamation/interrogation symbols, (6) interrogative sentences, (7) 'thank' mentions, (8) words that are not in WORD2VEC's Google News vocabulary. Also, (9) average number of tokens, and (10) word type-to-token ratio.

META FEATURES. (1) is the person answering the question the one who asked it; (2) reciprocal rank of comment c_m^i in the thread of q_i, i.e., $1/m$;

---

[2]BLEU FEATURES and BLEU COMPONENTS (Guzmán et al., 2016a,b) are ported from an MT evaluation framework (Guzmán et al., 2015; Guzmán et al., 2017) to cQA.

	batch	dropout	reg. str	inter. layer	task-spec. layer
A	16	0.3	0.001	10	125
B	25	0.2	0.05	5	75
C	32	0.3	0.0001	15	50

Table 1: Best setting for DNNs, as found on DEV.

(3) reciprocal rank of c_m^i in the list of comments for q, i.e., $1/[m+10\times(i-1)]$; and (4) reciprocal rank of question q_i in the list for q, i.e., $1/i$.

5 Data and Settings

We experiment with the data from SemEval-2016 Task 3 (Nakov et al., 2016b). Consistently with our notation from Section 3, it features three subtasks: subtask A (i.e., whether a comment c_m^i is a good answer to the question q_i in the thread), subtask B (i.e., whether the retrieved question q_i is related to the new question q), and subtask C (i.e., whether the comment c_m^i is a relevant answer for the new question q). Note that the two main subtasks we are interested in are B and C.

DNN Setting. We preprocess the data using min-max scaling. We use RMSprop[3] for learning, with parameters set to the values suggested by Tieleman and Hinton (2012). We use up to 100 epochs with patience of 25, rectified linear units (ReLU) as activation functions, l_2 regularization on weights, and dropout (Srivastava et al., 2014) of hidden units. See Table 1 for more detail.

CRF Setting. For the CRF model, we initialize the node-level weights from the output layer weights of the DNNs, and we set the edge-level weights to 0. Then, we train using RMSprop with loopy BP. We regularize the node parameters according to the best settings of the DNN: 0.001, 0.05, and 0.0001 for A, B, and C, respectively.

6 Results and Discussion

Below, we first present the evaluation results using DNN models (Section 6.1). Then, we discuss the performance of the joint models (Section 6.2).

6.1 Results for the DNN Models

Table 2 shows the results for our individual DNN models (rows in boldface) for subtasks A, B and C on the TEST set.
We report three ranking-based measures that are commonly accepted in the IR community: mean average precision (MAP), which was the official

---

Subtask A

System	MAP	AvgRec	MRR
Random order	52.80	66.52	58.71
Chronological order	59.53	72.60	67.83
ConvKN (second at SE-2016)	77.66	88.05	84.93
Kelp (best at SE-2016)	79.19	88.82	86.42
DNN$_A$ (subtask A network)	**76.20**	**86.52**	**84.95**

Subtask B

System	MAP	AvgRec	MRR
Random order	46.98	67.92	50.96
IR order	74.75	88.30	83.79
ConvKN (second at SE-2016)	76.02	90.70	84.64
UH-PRHLT (best at SE-2016)	76.70	90.31	83.02
DNN$_B$ (subtask B network)	**76.27**	**90.27**	**83.57**
DNN$_B$ + A gold labels	76.10	89.96	83.62
DNN$_B$ + C gold labels	77.19	90.78	83.73
DNN$_B$ + A and C gold labels	77.12	90.71	83.73

Subtask C

System	MAP	AvgRec	MRR
Random order	15.01	11.44	15.19
IR+Chron. order	40.36	45.97	45.83
Kelp (second at SE-2016)	52.95	59.27	59.23
SUper team (best at SE-2016)	55.41	60.66	61.48
DNN$_C$ (subtask C network)	**54.24**	**58.30**	**61.47**
DNN$_C$ + A gold labels	61.14	66.67	66.86
DNN$_C$ + B gold labels	56.29	61.11	62.67
DNN$_C$ + A and B gold labels	63.49	71.16	68.19

Table 2: Results for our DNN models on all cQA subtasks, compared to the top-2 systems from SemEval-2016 Task 3. Inter-subtask dependencies are explored using gold output labels.

evaluation measure of SemEval-2016, average recall (AvgRec), and mean reciprocal rank (MRR).

For each subtask, we show two baselines and the results of the top-2 systems at SemEval. The first baseline is a random ordering of the questions/comments, assuming no knowledge about the subtask. The second baseline keeps the chronological order of the comments for subtask A, of the question ranking from the IR engine for subtask B, and both for subtask C.

We can see that the individual DNN models for subtasks B and C are very competitive, falling between the first and the second best at SemEval-2016. For subtask A, our model is weaker, but, as we will see below, it can help improve the results for subtasks B and C, which are our focus here.

Looking at the results for subtask C, we can see that sizeable gains are possible when using gold labels for subtasks A and B as features to DNN$_C$, e.g., adding gold A labels yields +6.90 MAP points.
Similarly, using gold labels for subtask B adds +2.05 MAP points absolute. Moreover, the gain is cumulative: using the two gold labels together

yields +9.25 MAP points. The same behavior is observed for the other evaluation measures. Of course, as we use gold labels, this is an upper bound on performance, but it justifies our efforts towards a joint multitask learning model.

6.2 Results for the Joint Model

Below we discuss the evaluation results for the joint model. We focus on subtasks B and C, which are the main target of our study.

Results for Subtask C. Table 3 compares several variants of the CRF model for joint learning, which we described in Section 3.2 above.

Row 1 shows the results for our individual DNN_C model. The following rows 2–4 present a *pipeline* approach, where we first predict labels for subtasks A and B and then we add these predictions as features to DNN_C. This is prone to error propagation, and improvements are moderate and inconsistent across the evaluation measures.

The remaining rows correspond to variants of our CRF model with different graph structures. Overall, the improvements over DNN_C are more sizeable than for the pipeline approach (with one single exception out of 24 cases); they are also more consistent across the evaluation measures, and the improvements in MAP over the baseline range from +0.96 to +1.76 points absolute.

Rows 5–8 show the impact of adding connections to subtasks A and B when solving subtask C (see Figure 2b). Interestingly, we observe the same pattern as with the gold labels: the A-C and B-C connections help individually and in combination, with A-C being more helpful. Yet, further adding A-B does not improve the results (row 8).

Note that the locally normalized joint model in Eq. 4 yields much lower results than the globally normalized CRF_{all} (row 8): 54.32, 59.87, and 61.76 in MAP, AvgRec and MRR (figures not included in the table for brevity). This evinces the problems with the conditional independence assumption and the local normalization in the model.

Finally, rows 9–12 explore variants of the best system from the previous set (row 7), which has connections between subtasks only. Rows 9–12 show the results when using subgraphs for A, B and C that are fully connected (i.e., for all pairs). We can see that none of these variants yields improvements over the model from row 7, i.e., the fine-grained relations between comments in the threads and between the different related questions

do not seem to help solve subtask C in the joint model. Note that our scores from row 7 are better than the best results achieved by a system at SemEval-2016 Task 3 subtask C: 56.00 vs. 55.41 on MAP, and 63.25 vs. 61.48 on MRR.

Results for Subtask B. Next, we present in Table 4 similar experiments, but this time with subtask B as the target, and we show some more measures (accuracy, precision, recall, and F_1).

Given the insights from Table 2 (where we used gold labels), we did not expect to see much improvements for subtask B. Indeed, as rows 2–4 show, using the pipeline approach, the IR measures are basically unaltered. However, classification accuracy improves by almost one point absolute, recall is also higher (trading for lower precision), and F_1 is better by a sizeable margin.

Coming to the joint models (rows 6–9), we can see that the IR measures improve consistently over the pipeline approach, even though not by much. The effect on accuracy-P-R-F_1 is the same as observed with the pipeline approach but with larger differences.[4] In particular, accuracy improves by more than two points absolute, and recall increases, which boosts F_1 to almost 60.

Row 5 is a special case where we only consider subtask B, but we do the learning and the inference over the set of ten related questions, exploiting their relations. This yields a slight increase in all measures; more importantly, it is crucial for obtaining better results with the joint models.

Rows 6–9 show results for various variants of the A-C and B-C architecture with fully connected B nodes, playing with the fine-grained connection of the A and C nodes. The best results are in this block, with increases over DNN_B in MAP (+0.61), AvgRec (+0.69) and MRR (+1.05), and especially in accuracy (+2.18) and F_1 (+11.25 points). This is remarkable given the low expectation we had about improving subtask B.

Note that the best architecture for subtask C from Table 3 (A-C and B-C with no fully connected B layer) does not yield good results for subtask B.

We speculate that subtask B is overlooked by the architecture, which has many more connections and parameters on the nodes for subtasks A and C (ten comments are to be classified for both subtask

---

[4] Note that we have a classification approach, which favors accuracy-P-R-F_1; if we want to improve the ranking measures, we should optimize for them directly.

#	System	Comments	MAP (Δ)	AvgRec (Δ)	MRR (Δ)
1	DNN_C	Subtask C network	54.24	58.30	61.47
2	DNN_{C+PA}	DNN_C with A predicted labels	55.21 (+0.97)	58.36 (+0.06)	62.69 (+1.22)
3	DNN_{C+PB}	DNN_C with B predicted labels	54.17 (-0.04)	58.17 (-0.13)	62.55 (+1.08)
4	$DNN_{C+PA+PB}$	DNN_C with A and B predicted labels	55.11 (+0.90)	58.69 (+0.39)	60.10 (-1.37)
5	CRF_{AC}	CRF with A-C connections	55.42 (+1.18)	58.69 (+0.39)	63.25 (+1.78)
6	CRF_{BC}	CRF with B-C connections	55.20 (+0.96)	58.87 (+0.57)	62.30 (+0.83)
7	CRF_{ACBC}	CRF with A-C and B-C connections	**56.00** (+1.76)	60.20 (+1.90)	**63.25** (+1.78)
8	CRF_{all}	CRF with all pairwise connections	55.81 (+1.57)	60.15 (+1.85)	62.68 (+1.21)
9	$CRF_{ACBC,Cf}$	CRF_{ACBC} with fully connected C	55.73 (+1.49)	59.77 (+1.47)	62.80 (+1.33)
10	$CRF_{ACBC,AfCf}$	CRF_{ACBC} with fully connected A and C	55.54 (+1.30)	59.86 (+1.56)	62.54 (+1.07)
11	$CRF_{ACBC,BfCf}$	CRF_{ACBC} with fully connected B and C	55.67 (+1.43)	**60.22** (+1.92)	62.80 (+1.33)
12	$CRF_{ACBC,f}$	CRF_{ACBC} with all layers fully connected	55.81 (+1.57)	60.15 (+1.85)	**63.25** (+1.78)

Table 3: Performance of the *pipeline* and of the joint learning models on subtask C. The best results for each measure are in bold, and the gains over the single neural network (DNN_C) are shown in parentheses.

#	System	Comments	MAP	AvgRec	MRR	Acc	P	R	F_1
1	DNN_B	Subtask B network	76.27	90.27	83.57	76.39	**89.53**	33.05	48.28
2	DNN_{B+PA}	DNN_B with A predicted labels	76.08	89.99	83.38	77.40	86.41	38.20	52.98
3	DNN_{B+PC}	DNN_B with C predicted labels	76.33	90.38	83.62	77.40	83.19	40.34	54.34
4	$DNN_{B+PA+PC}$	DNN_B with A and C predicted labels	76.43	90.34	83.62	77.11	78.74	42.92	55.56
5	CRF_{Bf}	CRF with fully connected B	76.41	90.34	83.81	77.00	84.62	37.76	52.23
6	$CRF_{ACBC,Bf}$	CRF_{ACBC} with fully connected B	**76.89**	90.87	84.19	77.86	76.00	**48.93**	**59.53**
7	$CRF_{ACBC,AfBf}$	CRF_{ACBC} with fully connected A and B	76.51	90.64	84.19	78.29	83.47	43.35	57.06
8	$CRF_{ACBC,BfCf}$	CRF_{ACBC} with fully connected B and C	76.87	**90.96**	84.44	77.86	78.68	45.92	58.00
9	$CRF_{ACBC,f}$	CRF_{ACBC} with all layers fully connected	76.25	90.38	**84.62**	**78.57**	81.20	46.35	59.02

Table 4: Performance of the *pipeline* and of the joint models on subtask B (best results in boldface).

A and C, while only one decision is to be made for the related question B).

Finally, note that our best results for subtask B are also slightly better than those for the best system at SemEval-2016 Task 3, especially on MRR.

7 Conclusion

We have presented a framework for multitask learning of two community Question Answering problems: question-question relatedness and answer selection. We further used a third, auxiliary one, i.e., finding the good comments in a question-comment thread. We proposed a two-step framework based on deep neural networks and structured conditional models, with a feed-forward neural network to learn task-specific embeddings, which are then used in a pairwise CRF as part of a multitask model for all three subtasks.

The DNN model has its strength in generating compact embedded representations for the subtasks by modeling interactions between different input elements.

On the other hand, the CRF is able to perform global inference over arbitrary graph structures accounting for the dependencies between subtasks to provide globally good solutions. The experimental results have proven the suitability of combining the two approaches. The DNNs alone already yielded competitive results, but the CRF was able to exploit the task-specific embeddings and the dependencies between subtasks to improve the results consistently across a variety of evaluation metrics, yielding state-of-the-art results.

In future work, we plan to model text complexity (Mihaylova et al., 2016), veracity (Mihaylova et al., 2018), speech act (Joty and Hoque, 2016), user profile (Mihaylov et al., 2015), trollness (Mihaylov et al., 2018), and goodness polarity (Balchev et al., 2016; Mihaylov et al., 2017). From a modeling perspective, we want to strongly couple CRF and DNN, so that the global errors are backpropagated from the CRF down to the DNN layers. It would be also interesting to extend the framework to a cross-domain (Shah et al., 2018) or a cross-language setting (Da San Martino et al., 2017; Joty et al., 2017). Trying an ensemble of neural networks with different initial seeds is another possible research direction.

References

Daniel Balchev, Yasen Kiprov, Ivan Koychev, and Preslav Nakov. 2016. PMI-cool at SemEval-2016 Task 3: Experiments with PMI and goodness polarity lexicons for community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, pages 844–850, San Diego, California, USA.

Alberto Barrón-Cedeño, Simone Filice, Giovanni Da San Martino, Shafiq Joty, Lluís Màrquez, Preslav Nakov, and Alessandro Moschitti. 2015. Thread-level information for comment classification in community question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, ACL-IJCNLP '15, pages 687–693, Beijing, China.

Daniele Bonadiman, Antonio Uva, and Alessandro Moschitti. 2017. Effective shared representations with multitask learning for community question answering. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, EACL '17, pages 726–732, Valencia, Spain.

Charles Chen and Razvan Bunescu. 2017. An exploration of data augmentation and RNN architectures for question ranking in community question answering. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, IJCNLP '17, pages 442–447, Taipei, Taiwan.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Giovanni Da San Martino, Salvatore Romeo, Alberto Barrón-Cedeño, Shafiq Joty, Lluís Màrquez, Alessandro Moschitti, and Preslav Nakov. 2017. Cross-language question re-ranking. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 1145–1148, Tokyo, Japan.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, pages 138–145, San Francisco, California, USA.

Simone Filice, Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2016. KeLP at SemEval-2016 Task 3: Learning semantic relations between questions and answers. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, pages 1116–1123, San Diego, California, USA.

Francisco Guzmán, Shafiq Joty, Lluís Màrquez, and Preslav Nakov. 2015. Pairwise neural machine translation evaluation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, ACL-IJCNLP '15, pages 805–814, Beijing, China.

Francisco Guzmán, Shafiq R. Joty, Lluís Màrquez, and Preslav Nakov. 2017. Machine translation evaluation with neural networks. *Computer Speech & Language*, 45:180–200.

Francisco Guzmán, Lluís Màrquez, and Preslav Nakov. 2016a. Machine translation evaluation meets community question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, ACL '16, pages 460–466, Berlin, Germany.

Francisco Guzmán, Preslav Nakov, and Lluís Màrquez. 2016b. MTE-NN at SemEval-2016 Task 3: Can machine translation evaluation help community question answering? In *Proceedings of the International Workshop on Semantic Evaluation*, SemEval '16, pages 887–895, San Diego, California, USA.

Doris Hoogeveen, Li Wang, Timothy Baldwin, and Karin M. Verspoor. 2018. Web forum retrieval and text analytics: A survey. *Foundations and Trends in Information Retrieval*, 12(1):1–163.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Shafiq Joty, Alberto Barrón-Cedeño, Giovanni Da San Martino, Simone Filice, Lluís Màrquez, Alessandro Moschitti, and Preslav Nakov. 2015. Global thread-level inference for comment classification in community question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, EMNLP '15, pages 573–578, Lisbon, Portugal.

Shafiq Joty 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*, ACL '16, pages 1746–1756, Berlin, Germany.

Shafiq Joty, Lluís Màrquez, and Preslav Nakov. 2016. Joint learning with global inference for comment classification in community question answering. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '16, pages 703–713, San Diego, California, USA.

Shafiq Joty, Preslav Nakov, Lluís Màrquez, and Israa Jaradat. 2017. Cross-language learning with adversarial neural networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning*, CoNLL '17, pages 226–37, Vancouver, Canada.

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

John Lafferty, 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, California, USA.

Tuan Manh Lai, Trung Bui, and Sheng Li. 2018. A review on deep learning techniques applied to answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, COLING '18, pages 2132–2144, Santa Fe, New Mexico, USA.

Alon Lavie and Michael Denkowski. 2009. The METEOR metric for automatic evaluation of machine translation. *Machine Translation*, 23(2–3):105–115.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluís Màrquez. 2016. Semi-supervised question retrieval with gated convolutions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '16, pages 1279–1289, San Diego, California, USA.

Xuezhe Ma 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*, ACL '16, pages 1064–1074, Berlin, Germany.

Todor Mihaylov, Daniel Balchev, Yasen Kiprov, Ivan Koychev, and Preslav Nakov. 2017. Large-scale goodness polarity lexicons for community question answering. In *Proceedings of the 40th International Conference on Research and Development in Information Retrieval*, SIGIR '17, Tokyo, Japan.

Todor Mihaylov, Georgi Georgiev, and Preslav Nakov. 2015. Finding opinion manipulation trolls in news community forums. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, CoNLL '15, pages 310–314, Beijing, China.

Todor Mihaylov, Tsvetomila Mihaylova, Preslav Nakov, Lluís Màrquez, Georgi Georgiev, and Ivan Koychev. 2018. The dark side of news community forums: Opinion manipulation trolls. *Internet Research*.

Todor Mihaylov and Preslav Nakov. 2016. SemanticZ at SemEval-2016 Task 3: Ranking relevant answers in community question answering using semantic similarity based on fine-tuned word embeddings. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, pages 879–886, San Diego, California, USA.

Tsvetomila Mihaylova, Pepa Gencheva, Martin Boyanov, Ivana Yovcheva, Todor Mihaylov, Momchil Hardalov, Yasen Kiprov, Daniel Balchev, Ivan Koychev, Preslav Nakov, Ivelina Nikolova, and Galia Angelova. 2016. SUper Team at SemEval-2016 Task 3: Building a feature-rich system for community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, pages 836–843, San Diego, California, USA.

Tsvetomila Mihaylova, Preslav Nakov, Lluís Màrquez, Alberto Barrón-Cedeño, Mitra Mohtarami, Georgi Karadjov, and James Glass. 2018. Fact checking in community forums. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, AAAI '18, pages 879–886, New Orleans, Louisiana, USA.

Tomas Mikolov, 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, Georgia, USA.

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

Kevin P. Murphy, 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, Sweden.

Preslav Nakov, Lluís Màrquez, and Francisco Guzmán. 2016a. It takes three to tango: Triangulation approach to answer ranking in community question answering. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, EMNLP '16, pages 1586–1597, Austin, Texas, USA.

Preslav Nakov, Lluís Màrquez, Alessandro Moschitti, Walid Magdy, Hamdy Mubarak, abed Alhakim Freihat, James Glass, and Bilal Randeree. 2016b. SemEval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, pages 525–545, San Diego, California, USA.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meting of the Association for Computational Linguistics*, ACL '02, pages 311–318, Philadelphia, Pennsylvania, USA.

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

Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In *Proceedings of International Joint Conference on Artificial Intelligence*, IJCAI '15, pages 1305–1311, Buenos Aires, Argentina.

Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, ACL-IJCNLP '15, pages 694–699, Beijing, China.

Darsh J Shah, Tao Lei, Alessandro Moschitti, Salvatore Romeo, and Preslav Nakov. 2018. Adversarial domain adaptation for duplicate question detection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '18, Brussels, Belgium.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas*, AMTA '06, pages 223–231, Cambridge, Massachusetts, USA.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, pages 455–465, Sofia, Bulgaria.

Nitish Srivastava, 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.

Ming Tan, Bing Xiang, and Bowen Zhou. 2015. LSTM-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.

T. Tieleman and G Hinton. 2012. *RMSprop*. COURSERA: Neural Networks

S. V. N. Vishwanathan, 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, Pennsylvania, USA.

Di Wang and Eric Nyberg. 2015. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, ACL-IJCNLP '15, pages 707–712, Beijing, China.

Pengwei Wang, Lei Ji, Jun Yan, Dejing Dou, Nisansa De Silva, Yong Zhang, and Lianwen Jin. 2018. Concept and attention-based cnn for question retrieval in multi-view learning. *ACM Trans. Intell. Syst. Technol.*, 9(4):41:1–41:24.

Yair Weiss. 2001. Comparing the mean field method and belief propagation for approximate inference in MRFs. In *Advanced Mean Field Methods*, pages 229–239, Cambridge, Massachusetts, USA. MIT Press.

Wei Wu, Xu SUN, and Houfeng WANG. 2018. Question condensing networks for answer selection in community question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, ACL '18, pages 1746–1755, Melbourne, Australia.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701.

Xiaoqiang Zhou, Baotian Hu, Qingcai Chen, Buzhou Tang, and Xiaolong Wang. 2015. Answer sequence learning with neural networks for answer selection in community question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, ACL-IJCNLP '15, pages 713–718, Beijing, China.