Semi-supervised and Unsupervised Categorization of Posts in Web Discussion Forums using Part-of-Speech Information and Minimal Features

Krish Perumal
Department of Computer Science
University of Toronto
Toronto, ON, M5S 3G4, Canada
krish@cs.toronto.edu

Graeme Hirst
Department of Computer Science
University of Toronto
Toronto, ON, M5S 3G4, Canada
gh@cs.toronto.edu

Abstract

Web discussion forums typically contain posts that fall into different categories such as *question*, *solution*, *feedback*, *spam*, etc. Automatic identification of these categories can aid information retrieval that is tailored for specific user requirements. Previously, a number of supervised methods have attempted to solve this problem; however, these depend on the availability of abundant training data. A few existing unsupervised and semi-supervised approaches are either focused on identifying only one or two categories, or do not discuss category-specific performance. In contrast, this work proposes methods for identifying multiple categories, and also analyzes the category-specific performance. These methods are based on sequence models (specifically, hidden Markov Models) that can model language for each category using both probabilistic word and part-of-speech information, and minimal manually specified features. The unsupervised version initializes the models using clustering, whereas the semi-supervised version uses few manually labeled forum posts. Empirical evaluations demonstrate that these methods are more accurate than previous ones.

1 Introduction

Web discussion forums are platforms where people converse with one another to collaboratively solve problems and discuss issues. These are useful for existing users who participate in the discussion; however, new users need to read the entire forum thread for obtaining a solution or a summary of all opinions. This problem becomes much more pronounced in cases where threads contain tens or hundreds of posts, and reading the entire thread becomes impractical[1]. In such cases, labeling the purpose of each post can guide the user towards useful posts (i.e., containing solutions) and away from trivial posts (i.e., containing feedback or off-topic discussions). Moreover, current information retrieval techniques return entire threads as results to search queries. But by being sensitized to these annotations, they can return targeted results containing only the relevant posts. Further, user-contributed information contained in these forums can be better structured and can contribute towards the development of domain-specific knowledge bases. With these motivations in mind, this work aims to automatically annotate each post in a discussion forum with its purpose in the conversation thread. Our methods are not tailored for a specific domain or tagset. However, to demonstrate the objective of this work, Table 1 shows an example thread in which all posts are manually tagged with their purpose in the conversation.

2 Related Work

Categorizing forum posts is closely related to the task of *dialogue act tagging*, which is defined as the identification of the meaning of an utterance at the level of illocutionary force (Stolcke et al., 2000); for example, an utterance could be identified as falling

[1]For example, the JeepForum thread http://www.jeepforum.com/forum/f15/mud-tires-119948/ contains more than 500 posts discussing popular brands of tires.

Post	Purpose
User 15JKU: Hey Guys, Im looking to get 35s tires with either 18s or 20s as it will be more of a daily driver and sometimes go mudding. My only concern is how they will perform in mud? Also, how loud would they for a daily driven jeep? Also, would A/T tires work for mudding?	*Question*
User mschi772: You need to more accurately convey what your true priorities are. You're asking for too much from one tire.	*Request for Clarification*
User 15JKU: Just asking if anyone knows how loud they are. My main concern is how they'll do on mud and if i should go with different tire.	*Clarification*
User mschi772: Nitto Trail Grapplers are a "classic" MT design. This is a very popular design for people who frequently go offroading but want to maintain some street manners.	*Solution*
User JcArnold: I've got 37" trails and they are not noisy. I don't know about mud but they are great tires in the rocks and snow.	*Solution*
User 15JKU: Thanks guys! Truly appreciate it.	*Feedback*

Table 1: Example forum thread manually tagged with each post's purpose in the conversation. (Adapted from: `http://www.jeepforum.com/forum/f15/tire-recommendations-3455674/`)

into one or more categories such as *question*, *solution*, *clarification*, *feedback*, *command*, *request*, etc.

Most previous work has concentrated on supervised machine learning methods (Catherine et al., 2012; Bhatia et al., 2012; Wang et al., 2010; Kim et al., 2010; Sondhi et al., 2010) using manually annotated data in order to predict the annotations of unseen data. Apart from being constrained by the requirement of manually annotated data for training, these methods are also limited in applicability to the domains they are trained on. In contrast, unsupervised methods overcome these drawbacks by identifying unlabeled clusters of data, each of which could potentially be mapped to a target category that one wants to identify. To the best of our knowledge, three unsupervised techniques have been previously proposed for categorization of posts in Web forums. Out of these, Deepak and Visweswariah (2014) identified only *answer* posts, and Cong et al. (2008) additionally extracted *question* posts. The more difficult task of identifying multiple categories was tackled only by Joty et al. (2011). They used a combination of HMMs and Gaussian Mixture Models (GMMs) in order to classify forum posts into 12 dialogue act categories. This model is similar to the content and conversation models used for other tasks by Ritter et al. (2010) and Barzilay and Lee (2004) respectively. In addition to the probability distribution of word *n*-grams, they use some structural features such as the chronological position of a post in the thread, the number of tokens in the post, and author identity. The motivation for this approach is that HMMs can model the sequential nature of dialogue acts well. For example, the fact that a *solution* is more likely to follow a *question*, as opposed to any other category, can be implicitly encoded in the HMMs. Our approach is inspired by the same idea.

One major drawback of unsupervised methods is that they often generate clusters unrelated to the target categories. For example, clustering of forum posts on the travel domain might lead to a cluster containing posts pertaining to New York City sightseeing alone. This cluster is irrelevant when the purpose is to find clusters of post categories such as *question*, *answer*, *feedback*, etc. Moreover, because the clusters are unlabeled, post-processing is necessary to map the clusters to the target categories. Semi-supervised methods can overcome the drawbacks of both unsupervised and supervised methods by using a minimal amount of labeled data (which is costly to obtain) and a large amount of unlabeled data (which is easily available). To our knowledge, there exist only two semi-supervised methods for categorization of posts in Web forums and they identify only *answer* posts. Catherine et al. (2013) employed the co-training framework, whereas Jeong et al. (2009) used domain adaptation from labeled spoken dialogue datasets by means of a sub-tree pattern mining algorithm. In experiments, we show that our methods outperform the former work; however, the unavailability of code and data prevents empirical comparison with the latter.

3 Proposed Methods

3.1 Conversation Model

As discussed previously, our models derive inspiration from the work of Joty et al. (2011). Our underlying model is the same but differs in several important details. Our conversation model is a Hidden Markov Model (HMM), in which hidden (unobserved) states correspond to post categories, and emissions (observed) correspond to bags of post n-grams. Here, a thread T_k consists of a sequence of category labels, and each category label C_i emits a bag of word n-grams N_i of the i^{th} chronological post in the thread. The learning algorithm (Algorithm 1) of the conversation model uses iterative Expectation Maximization (EM) to maximize the expected probability of a post given a state, repeating until convergence of the sum of all observation probabilities. During the expectation step (E-step), a word n-gram language model is constructed for each state. Using this state-specific language model, the emission probability of an observation (or post) can be calculated. During the maximization step (M-step), the most likely state sequence is calculated using Viterbi algorithm. The language model for each state is constructed using smoothed n-gram frequency counts (using a smoothing parameter, δ_1). The parameter, $lmType$, determines the use of either unigrams or bigrams. The initial state probabilities and state transition probabilities are estimated using smoothed frequency counts of initial states and state transitions respectively (using a smoothing parameter of δ_2). The calculation of these estimates is based on work by Barzilay and Lee (2004) (and are different from those of Joty et al. (2011)). A more detailed explanation of this model is made available by Perumal (2016).

In the HMM, the probability of a post P_i, given a state S_k, is calculated as a categorical probability of its word n-grams, as shown in Equation 1.

$$p(P_i|S_k) = \prod_j p(W_{i,j}|L_k) \tag{1}$$

where $W_{i,j}$ is the j^{th} (in no particular order) word n-gram in post P_i, and L_k is the language model for state S_k.

3.1.1 Unsupervised Version

In the unsupervised version, the prior probabilities of the model are derived from a two-step process (based on the work of Barzilay and Lee (2004)): (i) every post is represented as a vector of word n-gram frequency counts, and (ii) the vectors are clustered using hierarchical clustering. The resultant cluster labels are used to calculate the frequency counts of initial HMM states and state transitions, and hence the corresponding probabilities. The priors are optionally calculated using an additional concept of *insertion states*. These are the states which contain a number of posts fewer than a fixed threshold, called *state size threshold*. This concept is used to account for small noise states that pertain to no meaningful target category. If used, all insertion states are merged into a single state, representing a noise state.

3.1.2 Semi-Supervised Version

Instead of using unsupervised clustering, we propose to derive the priors (i.e., the language model, the initial state probabilities and state transition probabilities) using smoothed frequency counts of post labels in few manually labeled threads. The rationale of this process is to form a better real-world estimate of the model parameters in the first EM iteration, and thereby reduce errors in the final predictions.

3.2 Conversation Model with Part-of-Speech Tags

Since the conversation models of Ritter et al. (2010) take only word n-gram language models into account, it is likely that they output clusters of posts that are topically related, without reflecting the posts' purpose or intention. To overcome this limitation, we enhance the plain conversation model by modeling HMM emissions partially from part-of-speech (POS) tags of words. This idea is based on the assumption that posts belonging to the same category are likely to be syntactically similar. For example, *question* posts are very likely to contain POS tags such as *WDT*, *WP*, *WP$*, and *WRB*[2]. Our model uses POS n-gram language models in addition to word n-gram language models, and calculates the HMM emission probability of a post given its

[2]These tags can be seen in the Penn Treebank project (Marcus et al., 1993).

state using a linear combination of both. Here, the probability of a post P_i, given a state S_k, is calculated as shown in Equation 2.

$$p(P_i|S_k) = \frac{\prod_j \left[\lambda \times p(W_{i,j}|L_k) + (1-\lambda) \times p(POS_{i,j}|PL_k)\right]}{Z}$$
$$0 \leq \lambda \leq 1 \quad (2)$$
$$Z = \sum_{i,k} \left[\prod_j \left[\lambda \times p(W_{i,j}|L_k) + (1-\lambda) \times p(POS_{i,j}|PL_k)\right]\right]$$

where $POS_{i,j}$ is the j^{th} (in no particular order) POS n-gram in post P_i, PL_k is the POS n-gram language model for state S_k, λ is the parameter that controls the proportion of probability arising from the word and POS language models (using $\lambda = 1$ is equivalent to the conversation model), and Z is the normalizing constant.

3.3 Conversation Model with Features

As a further enhancement to the conversation models, we incorporate discriminative features that might be useful for generating clusters that better represent the desired categories. For example, the chronological position of a post in a thread might be a useful feature, because a post is more likely to be a *question* if it is the first post in a thread as opposed to any other position. The following features are used: post position, post length, presence of question mark(s) (*?*, *???*, etc.) in current and preceding post, presence of *thank* or *thanks*, presence of *same* or *similar*, presence of *did*, presence of exclamation mark(s) (*!*, *!!!*, etc.), average cosine similarity with other posts in thread, cosine similarity with initial post, current post's author identity, whether previous post is by same author, number of previous posts by current author, and total number of posts by current author. All feature values are discretized. Joty et al. (2011) also use a few specific features in their model, but our approach is more general and can accommodate a variable number of features. The probability of a post P_i, given a state S_k, is calculated as shown in equation 3.

$$p(P_i|S_k) = \prod_j p(W_{i,j}|L_k) \prod_f p(F_{i,f}|FL_k) \quad (3)$$

where $F_{i,f}$ is the f^{th} (in no particular order) discrete-valued feature in post P_i, and FL_k is the feature model for state S_k.

Ubuntu (Bhatia et al., 2012) Domain: Computer technical Tagset: *Question*, *Repeat Question*, *Clarification*, *Solution*,*Further Details*, *Positive Feedback*, *Negative Feedback*, *Spam* Number of threads: 100
TripAdvisor-NYC (Bhatia et al., 2012) Domain: Travel Tagset: Same as **Ubuntu** Number of threads: 100
Apple (Catherine et al., 2012) Domain: Computer technical Tagset: *Answer* Number of threads: 300 labeled and 140,000 unlabeled

Table 2: Discussion forum datasets used in the current work's experiments.

3.4 Mapping of Clusters to Categories in Unsupervised Methods

Unsupervised methods output cluster labels for each post, not a specific category label. In order to pair them with an observed category label, a one-to-one mapping is obtained using the Kuhn-Munkres algorithm for maximal weighting in a bipartite graph (Kuhn, 1955; Munkres, 1957). The nodes of the graph correspond to the predicted cluster labels and gold labels, and the weights correspond to the number of overlapping posts between them. In this procedure, one set of disjoint nodes of the bipartite graph corresponds to the set of predicted cluster labels, and the other set corresponds to the set of manually obtained gold labels. The weight of an edge from cluster label c to gold label g is calculated as the number of posts which are predicted as c and also have a gold label g. Joty et al. (2011) follow the same procedure.

4 Experiments

4.1 Datasets

For our experiments, we use forum datasets from previous work. Details of their tagsets, sizes, and domains are listed in Table 2.

Algorithm 1 Conversation model

Input: A list of threads T, each containing a list of posts P (in chronological order)
Parameters: *initialNumClusters*, *mergeInsertionStates*, *stateSizeThreshold*, *maxNumIterations*, *lmType*, δ_1, δ_2
Output: A list of cluster labels CL for each post in each thread (in the order of the input)

for all thread T_x **do**
  for all post $P_{x,y} \in T_x$ **do**
    $V_{x,y} := vectorize(P_{x,y})$ // $V_{x,y}$ is the vector of post $P_{x,y}$
  end for
end for
$ICL := cluster(V, initialNumClusters)$ // ICL is the list of initial cluster labels for each post ($ICL_{x,y}$ is the initial cluster label for post $P_{x,y}$ in thread T_x)
$S := ICL$ // S is the list of states for all posts; at this step, it is the same as the initial cluster labels
for $n = 1 \rightarrow maxNumIterations$ **do**
  if *mergeInsertionStates* is *true* **then**
    $[S, numStates] := merge\_small\_states(S, stateSizeThreshold)$
  end if
  for $i = 1 \rightarrow numStates$ **do**
    $SP_i = \emptyset$
    for all state $S_{x,y}$ **do**
      if $S_{x,y} = i$ **then**
        $SP_i := SP_i \cup P_{x,y}$ // SP_i is the set of all posts that belong to state i
      end if
    end for
    $L_i := language\_model(SP_i, lmType, \delta_1)$
  end for
  for $i = 1 \rightarrow numStates$ **do**
    $init\_counts_i := \Sigma_{T_x} \mathbb{1} S_{x,1} = i$ // $S_{x,1}$ is the state of the first post in thread T_x
  end for
  for $i = 1 \rightarrow numStates$ **do**
    $\pi_i := (init\_counts_i + \delta_2) / (\Sigma_k (init\_counts_k) + \delta_2 \times numStates)$ // π_i is the probability that initial state is i
  end for
  for $i = 1 \rightarrow numStates$ **do**
    for $j = 1 \rightarrow numStates$ **do**
      $trans\_counts_{i,j} := \sum_{T_x} \sum_{a=1}^{|T_x|-1} \mathbb{1} S_{x,a} = i, S_{x,a+1} = j$
    end for
  end for
  for $i = 1 \rightarrow numStates$ **do**
    for $j = 1 \rightarrow numStates$ **do**
      $\phi_{i,j} := (trans\_counts_{i,j} + \delta_2) / (\Sigma_{k,l} (trans\_counts_{k,l}) + \delta_2 \times numStates^2)$ // $\phi_{i,j}$ is the probability of transitioning from state i to state j
    end for
  end for
  $S := Viterbi\_algorithm(\pi, \phi, L)$
  if sum of observation probabilities converged **then**
    break
  end if
end for
$CL := S$

4.2 Preprocessing and Configuration Parameters

Initially, all forum posts were tokenized by sentence and word, followed by POS tagging and stemming — all using Stanford CoreNLP Toolkit (Manning et al., 2014). Stopword removal was found to degrade performance; hence, it was not used. Forum conversations often consist of informal English language text, along with the use of domain-specific abbreviations and non-standard special characters such as ellipses and emoticons. Hence, some errors are introduced in all the previous steps. However, no effort was made to overcome them, and this is accepted as a limitation of the current work.

All methods require the conversion of posts to vectors of n-grams. For this purpose, we experimented with both unigrams and bigrams, and the former was found to produce better performance. The maximum number of iterations of Expectation Maximization was set to 100, which was sufficient because all experimental runs were completed in fewer than 100 iterations. The values of both smoothing parameters (i.e., δ_1 and δ_2) were varied in the range of 10^{-1} to 10^{-9}. Subsequently, 10^{-2} and 10^{-9} were found to be the best values for δ_1 and δ_2 respectively. The value of the POS model's λ was varied between 10^{-6} and $1-10^{-6}$, and the value of 0.999 was found to be the best. Since the unigram/bigram vocabulary size is much larger than the POS tag vocabulary size, the former probability distribution is much more fine-grained. For example, each word unigram's probability value in the NYC dataset is of the order of 10^{-4} (since the unigram vocabulary size is 5000), whereas each POS unigram's probability value is of the order of 10^{-2} (since the POS vocabulary size is 42). So, the value of 0.999 for word unigrams and 0.001 for POS unigrams can be viewed as a scaling factor to ensure that both contribute almost equally towards discriminating between post categories. The parameters, $initialNumClusters$ and $stateSizeThreshold$, directly affect the resulting number of clusters. In all experimental runs, both these parameters were varied in the range of 1 to 100, and those which did not output the desired number of clusters (i.e., number of distinct gold labels) were ignored. In each case, different parameter values were best suited; however, only the best performing results are reported.

For semi-supervised methods, experiments were carried out in a randomized n-fold cross-validation setup. The dataset was randomly divided (by sampling from the uniform distribution) into n equal-sized folds, and the experiment was run n times. In each run, one fold was used for initializing the priors of the models, and the remaining $n-1$ folds were used for evaluation. In the case of language models, different datasets benefited from using one of word/POS unigram or bigram models. Hence, experiments were run using both, and results are reported for the better performing alternative.

4.3 Baselines

The *random baseline* randomly assigns category labels to every post (by sampling from the uniform distribution). The *majority baseline* assigns the most commonly occurring gold category label to every post, which is *solution* for all the datasets used in this work. Two other baselines are heuristic in nature, and are both based on the assumption that the first post in the thread is very likely to be a *question*. The first of these, called *question-solution heuristic 1*, assigns *question* to the first post in the thread, *spam/other* to the last post, and *solution* to the rest. It assumes that the last post in the thread is very likely to be unrelated to the main thread topic and that many of the preceding posts are likely to be *solution*. The second heuristic baseline, called *question-solution heuristic 2*, assigns *question* to the first post in the thread, *solution* to the second post, and *spam/other* to the rest. It assumes that the second post is very likely to be a *solution* in direct response to the first *question* post, and many of the following posts are likely to be *spam/other*.

4.4 Main Results

Table 3 lists the accuracy values from experiments using all possible combinations of the implemented models for the Ubuntu and NYC datasets. For the unsupervised methods, the mean 1-to-1 accuracy values are reported using the procedure described in section 3.4. For the baselines and the semi-supervised methods, the reported accuracy values are averages over all categories.

	Accuracy (%)	
Method	**Ubuntu**	**NYC**
Baselines		
Random	78.11	77.71
Majority	85.37	87.91
Problem-Solution Heuristic 1	87.71	89.06
Problem-Solution Heuristic 2	82.88	80.69
Unsupervised		
HMM+Mix	83.30	87.91
CONV	88.85	90.06
CONV + POS	88.85	90.06
CONV + FEAT	89.26	90.83
CONV + POS + FEAT	89.26	90.83
Semi-Supervised		
HMM+Mix	82.33	86.91
CONV	88.45	90.74
CONV + POS	88.29	90.86
CONV + FEAT	89.08	91.31
CONV + POS + FEAT	89.10	91.72

Table 3: Experimental results for the Ubuntu and NYC datasets using all the possible combinations of models in both unsupervised and semi-supervised settings (CONV: Conversation model; POS: Part-of-speech tags; FEAT: Features).

Joty et al. (2011) reported results of their best performing HMM+Mix model for dialogue act classification on email and forum thread datasets. The code and datasets are not available to other researchers; hence, we implemented the HMM+Mix, while also accommodating the additional features that we used for our methods.

Our unsupervised methods beat all the baselines, in contrast to Joty et al. (2011)'s HMM+Mix, which beats only the random baseline. The use of both POS tags and features results in the best overall performance, whereas the use of POS tags does not make any difference in performance. The semi-supervised adaptation of the existing HMM+Mix model outperforms only the random baseline. However, all of our semi-supervised methods beat all the baselines. In this case, the sole use of POS tags or features results in improved performance. But the use of both in combination leads to the best performance overall. Specifically, for the Ubuntu dataset, the best average accuracy value is 89.10%. In case of the NYC dataset, the corresponding best value is 91.72%. The corresponding absolute accuracy values are 56.40% and 66.86% respectively. The reported results using semi-supervised methods are averages over 10 runs of a randomized 10-fold cross-validation setup.

Method	**P**	**R**	**F_1**
Catherine et al. (2013)	0.57	0.84	0.68
CONV + POS + FEAT	0.66	0.73	0.69

Table 4: Experimental results comparing the performance of an existing semi-supervised answer extraction method with our best semi-supervised method (i.e., conversation model with POS tags and features).

Category	**P**	**R**	**F_1**
Question	83.81	73.95	78.57
Repeat Question	0.00	0.00	0.00
Clarification	78.57	22.45	34.92
Further Details	40.00	8.51	14.04
Solution	66.05	96.56	78.44
+ve Feedback	0.00	0.00	0.00
-ve Feedback	60.78	30.10	40.26
Junk	33.33	4.00	7.14

Table 5: Experimental results of semi-supervised conversation model with POS tags and features for one of the folds in a 10-fold cross-validation setup using the NYC dataset.

Catherine et al. (2013) reported the performance of their semi-supervised *answer* extraction approach on 300 labeled threads of the Apple discussion forums dataset. They trained using only three training threads; however, the identities of these three are not known. The code is also unavailable. Hence, the methods can only be compared indirectly. For the methods of Catherine et al. (2013), precision, recall and F_1-measure values are obtained from their paper. The same values are reported for our best method (i.e., the semi-supervised conversation models with POS tags and features), using a 100-fold cross-validation setup; i.e., out of 300 labeled threads, 3 were used for training, and 297 were used for testing in each fold. Table 4 shows that our method performs better in terms of F_1-measure and precision.

4.5 Category-wise Performance and Error Analysis

Table 5 shows the category-wise performance of one of the runs of 10-fold cross-validation for the NYC dataset using the semi-supervised conversation

		Predicted							
		Q	**RQ**	**C**	**FD**	**S**	**F+**	**F-**	**J**
Actual	**Q**	104	0	5	0	7	0	2	1
	RQ	1	0	0	0	1	0	0	0
	C	2	0	11	0	40	0	0	2
	FD	16	0	1	1	20	0	12	1
	S	10	0	1	1	374	0	6	10
	F+	0	0	0	0	0	0	0	0
	F-	21	0	1	1	35	0	43	2
	J	2	0	1	0	17	0	11	10

Table 6: Confusion matrix of the semi-supervised conversation model with POS tags and features, for one of the folds in a 10-fold cross-validation setup using the NYC dataset (Q: Question; RQ: Repeat Question; C: Clarification; FD: Further Details; S: Solution; F+: Positive Feedback; F-: Negative Feedback; J: Junk).

model with POS tags and features. Table 6 shows the confusion matrix of the same experiment. The confusion matrix for the Ubuntu dataset is similar. Predictions of *question*, *solution*, *clarification* and *negative feedback* are the best in terms of precision values; however, the recall values of the two latter categories are not practically useful. The most common error is the prediction of a non-*solution* category as *solution*, indicating a bias towards predicting the majority category. Overall, the predictions of minority categories are not practically useful, because they were less accurate than the predictions using the random baseline. Specifically, there are no predictions of *repeat question* and *positive feedback*, owing to the minuscule number of posts with these labels. Since previous literature ignores the analysis of category-wise performance altogether, a direct comparison is not possible. But this confusion matrix indicates a major weakness of current semi-supervised and unsupervised approaches in classifying minority categories.

5 Conclusions and Future Work

Our experimental results indicate that our unsupervised methods are not adequate for tackling a task as complex as forum post categorization. However, they are able to capture some useful sequential dependencies, as observed from the fact that they outperformed the random and majority baselines. Also, knowledge of POS tags and simple textual features provided more context for classification, and thus enabled the technique to classify more accurately. The novel proposal of incorporating a few labeled examples for initializing the model priors led to better performance than the *question-solution heuristic* baselines in most cases. Our experiments demonstrate that these methods perform better than previous methods. Prediction of *question* and *solution* categories were the most accurate, followed by *clarification* and *negative feedback*. However, predictions of the minority categories are not accurate enough to be practically useful.

Discussion forum posts often contain multiple dialogue categories, i.e., a post could start with some sentence(s) mentioning a *solution* to a previous *question*, and end with some sentence(s) posing a new *question*. Such cases can be tackled by employing a 2-tier hierarchical HMM which models the transition between sentence categories within a single post (in addition to the higher-level post category transitions). However, this proposal is dependent on the availability of datasets that are annotated by category at the sentence level. The lack of knowledge of long-range dependencies between different categories is another drawback of current methods. Consequently, they are unable to learn that a post cannot be classified as *solution*, without any *question* post before it. This problem can be addressed by using higher-order Markov chains, but this would lead to much greater run-time and space complexity as well as specially tailored algorithms for inference. Instead, the use of heuristics to flag certain categories, based on prior post categories in the thread, could resolve this problem more efficiently. Moreover, effort should be made to balance the distribution of categories that are used for initializing and training the methods. We leave these ideas for exploration in future work.

Acknowledgments

This research was supported by the Natural Sciences and Engineering Research Council of Canada under the Engage grant (no. EGP 477227-14). We thank Afsaneh Fazly and Mohamed Abdalla for their valuable contributions, and thank the anonymous reviewers for their constructive feedback.

References

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications

to generation and summarization. In *Proceedings of the 2nd Human Language Technology Conference and Annual Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 113–120.

Sumit Bhatia, Prakhar Biyani, and Prasenjit Mitra. 2012. Classifying user messages for managing Web forum data. In *Proceedings of the 15th International Workshop on the Web and Databases (WebDB'12)*, pages 13–18.

Rose Catherine, Amit Singh, Rashmi Gangadharaiah, Dinesh Raghu, and Karthik Visweswariah. 2012. Does similarity matter? The case of answer extraction from technical discussion forums. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 175–184.

Rose Catherine, Rashmi Gangadharaiah, Karthik Visweswariah, and Dinesh Raghu. 2013. Semi-supervised answer extraction from discussion forums. In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 1–9.

Gao Cong, Long Wang, Chin-Yew Lin, Young-In Song, and Yueheng Sun. 2008. Finding question-answer pairs from online forums. In *Proceedings of the 31st Annual International ACM SIGIR Conference*, pages 467–474.

P Deepak and Karthik Visweswariah. 2014. Unsupervised solution post identification from discussion forums. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 155–164.

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

Shafiq Joty, Giuseppe Carenini, and Chin Yew Lin. 2011. Unsupervised modeling of dialog acts in asynchronous conversations. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1807–1813.

Su Nam Kim, Li Wang, and Timothy Baldwin. 2010. Tagging and linking web forum posts. In *Proceedings of the 14th Conference on Computational Natural Language Learning (CoNLL)*, pages 192–202.

Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):313–330.

James Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38.

Krish Perumal. 2016. Semi-supervised and unsupervised methods for categorizing posts in web discussion forums. Master's thesis, University of Toronto.

Alan Ritter, Colin Cherry, and Bill Dolan. 2010. Unsupervised modeling of Twitter conversations. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 172–180.

Parikshit Sondhi, Manish Gupta, ChengXiang Zhai, and Julia Hockenmaier. 2010. Shallow information extraction from medical forum data. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pages 1158–1166.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–373.

Li Wang, Su Nam Kim, and Timothy Baldwin. 2010. Thread-level analysis over technical user forum data. In *Proceedings of the Australasian Language Technology Association Workshop*, pages 27–31.