Analyzing Conversations using Rich Phrase Patterns

Bin Zhang, Alex Marin, Brian Hutchinson, Mari Ostendorf

Department of Electrical Engineering, University of Washington, Seattle, WA 98195, USA

{binz,iskander,brianhutchinson,mo}@ee.washington.edu

***Abstract*—Individual words are not powerful enough for many complex language classification problems. N-gram features include word context information, but are limited to contiguous word sequences. In this paper, we propose to use phrase patterns to extend n-grams for analyzing conversations, using a discriminative approach to learning patterns with a combination of words and word classes to address data sparsity issues. Improvements in performance are reported for two conversation analysis tasks: speaker role recognition and alignment classification.**

I. Introduction

Much work in language processing has focused on understanding the meaning of individual utterances, but recent work is considering conversations as a whole, in both spoken and online written forms. Analysis of conversation dynamics reveals information about the participants' social positioning and their relations to each other. In this paper, we explore novel lexical feature extraction methods in the context of detection of social phenomena in multi-party conversations.

Previous work shows the effectiveness of word features in text topic categorization [1], for which the bag-of-words model works well [2]. Recent work in sentiment analysis [3], [4], [5], [6] also use word features to classify positive and negative sentiment, although heuristics are typically introduced to handle phenomena such as negation [3]. For many higher-level tasks, single words are not powerful enough, since the meaning of a word can change based on its context. One simple way to capture local word context is by using higher-order n-gram features. Phrase patterns further generalize n-grams by allowing gaps between words, thus capturing longer-range context. Previous research has found a benefit in using phrase patterns as features in various applications, including subjectivity sentence detection [7], grammatical errors in texts written by second language learners [8], speaker role recognition [9], and recognition of sarcastic sentences in Twitter and Amazon product reviews [10], [11].

Prior work using phrase patterns uses either high frequency or hand-crafted rules to select the patterns, including the use of linguistically motivated word classes. Two advances in learning phrase patterns are introduced here, both motivated by problems of sparse training data. Phrase patterns (like n-grams) lead to high-dimensional feature spaces, which can result in overtraining for most learning algorithms. The problem can be mitigated with regularization, to an extent; however, even regularization has its limits when the number of features is large relative to the training set size. Therefore, we develop a method for replacing frequency-based pruning of phrase patterns with mutual information-based pruning, filtering out non-discriminative features. Another mechanism for dealing with sparse data is to use word classes. By grouping words into classes, we can not only obtain better estimation of word counts, but also make the features extend to unseen data, when the classes are matched to the task. Of course, word classes alone have reduced discriminative power compared to words. The challenge is in picking the right word classes, including when to use words vs. classes. Thus, our second contribution is in extending the phrase selection algorithm to automatically learn when to use word classes in phrase patterns, by embedding the classes in the word sequences provided to the learning algorithm. To assess the utility of this approach, the phrase pattern learning method is applied with a variety of word classes to two conversation analysis tasks, namely speaker role recognition and alignment classification.

Speaker role recognition assigns labels to each conversation participant. For example, our work on broadcast conversations (talk shows), with conversation participants labeled as one of five categories: host, guest participant, audience participant, reporting participant, and other. Previous work on radio/TV broadcasts has investigated the use of both acoustic and lexical cues for speaker role recognition, by supervised learning [12] and unsupervised clustering [13]. Liu [14] used only lexical cues, e.g. n-gram features, to tag the roles in the sequence of turns. Garg et al. [15] combined lexical and social network analysis for role recognition in meetings. Wang et al. [16] employed both lexical and structural features to tag speaker roles in broadcast conversations. Our previous work [9] examined the use of phrase patterns in unsupervised speaker role clustering, showing superior performance compared to n-gram features. The present work shows that phrase pattern features are also helpful in a supervised setting and suggests a method for mining discriminative phrase patterns.

Our second task, alignment classification, involves detecting supporting or dissenting moves with regard to another discussion participant. Previous work has investigated agreement/disagreement classification in meeting speech [17], [18], [19] and broadcast conversations [16], [20]. We consider online conversations in Wikipedia discussion pages, which are discussion forums associated with individual Wikipedia articles where editors discuss changes to an article. This paper shows that alignment classification can also be improved by the use of phrase patterns.

II. Phrase Patterns with Words and Word Classes

We define phrase patterns by expanding upon the concept of sequential patterns used in data mining, which have been applied to areas such as customer buying strategy extraction [21]. A sentence contains an ordered list of words. Associated with each word there are a number of attributes, such as the part-of-speech (POS) and polarity of the word. For a given word in the sentence, we create an *itemset* s as a set that contains the word identity and various attributes for this word; a sentence can then be considered as a sequence of itemsets $(s_1, s_2, \ldots, s_L)$, with each itemset containing a word and all its attributes. A phrase pattern is a subsequence of itemsets $(t_1, t_2, \ldots, t_M)$, and it is matched to a sentence if there exist

$$1 \leq i_1 < i_2 < \cdots < i_L \leq L$$

such that itemsets $t_j \subseteq s_{i_j},\ j = 1, 2, \ldots, M$. An example of a phrase pattern matched by two different sentences is illustrated in figure 1, with the matching subsequence indicated by boxed words and classes.

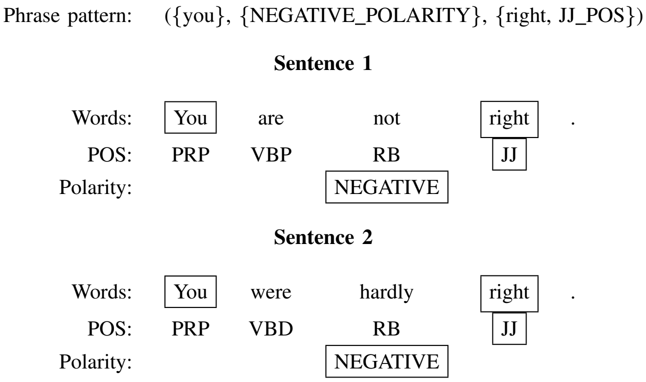
Phrase pattern: ({you}, {NEGATIVE_POLARITY}, {right, JJ_POS})
Sentence 1
Words: You are not right .
POS: PRP VBP RB JJ
Polarity: NEGATIVE
Sentence 2
Words: You were hardly right .
POS: PRP VBD RB JJ
Polarity: NEGATIVE

Fig. 1. Example of a phrase pattern matched by two sentences. We expand the words in sentences by word classes (POS and word polarity shown). Boxed elements are matches to the phrase pattern.

Phrase patterns are potentially helpful for natural language processing for several reasons. First, they extend n-grams by allowing gaps between words, therefore being able to model long-span behavior by skipping disfluencies and infrequent named entities. Second, richer information can be encapsulated in phrase patterns, by using a variety of word attributes. In particular, using word classes as word attributes allows us to produce discriminative but generic phrase patterns. For example, in figure 1, the phrase pattern ({you}, {NEGATIVE_POLARITY}, {right, JJ_POS}) matches many sentences with dissent. However, that pattern would not match the sentence *you don't have the right* which is not necessarily expressing dissent, since the word "right" would have a different POS tag (NN). The flexibility given by the use of words, word classes, or mixtures of words and word classes as itemsets enables us to create discriminative features targeted for specific tasks and data conditions.

III. Mining Discriminative Phrase Patterns

There has been extensive work in mining frequent sequential patterns, where the objective is to find the sequential patterns that are contained in more than a predefined number of sequences in a database. We can adapt these methods to mine phrase patterns, if we consider phrase patterns as sequential patterns, sentences as sequences, and text corpora as sequence databases. Efficient frequent pattern mining algorithms have been proposed, including *PrefixSpan* [22] (algorithm 1) and *CloSpan* [23]. These algorithms are based on recursive pattern growing, and efficiency is achieved by utilizing the fact that when the frequency of a sequential pattern is below the threshold, there is no need to continue searching for extending patterns as they have even lower frequency.

```
Algorithm 1: PrefixSpan($D, \rho, f$)
Input: A sequence database $D$, a prefix pattern $\rho$, the
       minimum frequency threshold $f$
Output: The complete set of sequential patterns $P$ in $D$
        with frequency greater than $f$
Initialize $P \leftarrow \emptyset$;
Scan all the sequences in $D$ once, find a set of items $A$
with frequencies of its elements no less than $f$, such that
  a) its element can be appended to $\rho$ to form a new
     sequential pattern; or
  b) its element can be assembled to the last itemset of $\rho$ to
     form a new sequential pattern;
for $a \in A$ do
    Create a new pattern $\rho'$ by appending $a$ to $\rho$;
    $P \leftarrow P \cup \{\rho'\}$;
    Create the $\rho'$-projected database $D'$ from $D$;
    Call PrefixSpan($D', \rho', f$) to obtain a set of patterns
    $B$;
    $P \leftarrow P \cup B$;
end
return $P$
```

The algorithm uses a subroutine to create a ρ'-projected database D' from D, where D' is the subset of D with only the sequences that match ρ'. Since PrefixSpan(D, ρ, f) is called recursively, this projection is crucial in reducing the number of sequences to be scanned in the following recursions.

Frequent sequential pattern mining algorithms are efficient for practical use, but they are not optimal for our conversation analysis tasks. Because not all frequent phrase patterns are discriminative, many phrase patterns mined using algorithm 1 are irrelevant to the task. On the other hand, we may miss some phrase patterns that do not appear frequently but are indeed discriminative. As a result, we need supervision in sequential pattern mining. The *ConSGapMiner* algorithm, introduced in [24], extends *PrefixSpan* and addresses the problem of discriminative sequential pattern mining to some extent. Let us assume that there are two databases, one containing only sequences with positive labels and the other only sequences with negative labels. *ConSGapMiner* mines the sequential patterns with frequency greater than δ in the positive database and less than α in the negative database, where δ and α are predefined thresholds. However, the drawback of this algorithm is that it does not easily extend to a multi-class setting. To overcome this, we propose to use an information theoretic criterion for discriminative sequential pattern mining.

A. The Mutual Information Criterion

Mutual information (MI) is the reduction of uncertainty in a random variable after observing another random variable. MI (which includes information gain) has been shown to be superior to many other feature selection criteria in text categorization tasks [25]. Suppose we have a data set with K classes, the MI between feature X and the class variable Y can be computed by

$$I(X;Y) = \sum_{x=0,1} \sum_{y=1}^{K} p(x,y) \log \frac{p(x,y)}{p(x)p(y)},$$

where X is 1 if the associated word or phrase is present and 0 if absent, and the probabilities are estimated from the data using maximum likelihood estimation. Features with $I(X;Y)$ greater than a fixed threshold are selected.

Different from the frequency criterion, the MI criterion is not prefix-monotonic, i.e., if a sequential pattern satisfies the MI criterion, its prefixes do not necessarily satisfy the criterion as well. Being unable to terminate the search preemptively, we lose the efficiency of the *PrefixSpan* algorithm. To overcome this, we rewrite the mutual information in the following form,

$$I(X;Y) = \sum_{x,y} p(x|y)p(y) \log \frac{p(x|y)}{\sum_{y'} p(x|y')p(y')},$$

where $p(y)$ is the prior class distribution, which is constant given the data. Given $p(y)$, $I(X;Y)$ is a convex function of $p(x|y)$ [26]. Let XE be the binary feature indicator of a pattern that extends X. Suppose we have already collected the statistics in the contingency table for estimating $p(x|y)$ and hence $I(X;Y)$, what can we infer about the mutual information of the extended pattern XE? Since the frequency of an extended pattern is always no greater than that of its prefix,

$$p(XE = 1|y) \leq p(X = 1|y),$$

we can derive an upperbound for the mutual information of all possible extended patterns

$$\max_{p(XE=1|y) \leq p(X=1|y)} I(XE;Y). \tag{1}$$

This quantity can then be used to determine if we should terminate the search for extended patterns: for a sequential pattern, if the upper bound computed by (1) is below the threshold θ, then the extended patterns will not be searched.

The maximizer of (1) satisfies

$$p(XE = 1|Y = i) = \left\{ \begin{array}{c} 0, \text{ or} \\ p(X = 1|Y = i) \end{array} \right. , \; i = 1, 2, \ldots, K.$$

This follows from the fact that maximization of a convex function on a compact convex set is always attained on the boundary of the constraints. For small K, (1) can be efficiently calculated.

B. Extended PrefixSpan

In this section, we propose an extended *PrefixSpan* algorithm to mine discriminative sequential patterns using the MI criterion. We use the MI criterion to filter sequential patterns, and use the upperbound (1) to decide whether to terminate. The procedure is outlined in algorithm 2. It is different from algorithm 1 in that, in the scanning of database, it finds not only the qualified patterns that satisfy Θ, but also the promising patterns for further mining. Algorithm 1 is a special case of algorithm 2, in which the qualified patterns are the same as the promising patterns. Algorithm 2 can be applied to a wider range of criteria Θ than mutual information. It does not require the criterion to be prefix-monotonic, as long as there is a way to efficiently compute the bounds.

Algorithm 2: ExtendedPrefixSpan(D, ρ, Θ)

```
Input: A sequence database $D$, a prefix pattern $\rho$, a
       criterion $\Theta$
Output: The complete set of sequential patterns $P$ in $D$
        satisfying criterion $\Theta$
Initialize $P \leftarrow \emptyset$;
Scan all the sequences in $D$ once, find a set of items $A$
satisfying criterion $\Theta$, and a set of items $B$ which have
promising extensions, such that either condition (a) or
condition (b) in algorithm 1 is satisfied;
for $a \in A$ do
    Create a new pattern $\rho'$ by appending $a$ to $\rho$;
    $P \leftarrow P \cup \{\rho'\}$;
end
for $b \in B$ do
    Create a new pattern $\rho'$ by appending $b$ to $\rho$;
    Create the $\rho'$-projected database $D'$ from $D$;
    Call ExtendedPrefixSpan($D', \rho', \Theta$) to obtain a set of
    patterns $C$;
    $P \leftarrow P \cup C$;
end
return $P$
```

When using criteria that are not prefix-monotonic, algorithm 2 needs to search more patterns than it outputs. However, it is more efficient than enumerating all sequential patterns and then apply post-filtering using Θ. In many cases, the enumeration of all sequential patterns is computationally prohibitive.

C. Phrase Pattern Implementation Notes

When a sequential pattern is matched against a sequence, gaps are allowed between itemsets, and the size of the gaps is unrestricted. This is counter-intuitive for text sequences, as for example a gap that spans a dozen words rarely makes sense. To address this problem, we enforce constraints such as $i_{j+1} - i_j \leq g$, where g is the maximum size gap allowed between the indices. The introduction of the gap size restriction complicates phrase pattern matching, which we address with an efficient algorithm based on bitset operations [24].

Intuitively, the matching of phrase patterns should be limited to a certain range, such as a sentence. The phrase pattern learning algorithms introduced above do not enforce that, unless each instance contains only one sentence, which may not be the case for many applications. Therefore, we explicitly use delimiters in the instances to limit the range of matching. A delimiter is a special token which does not take any room in the sequence. However, a phrase pattern can never skip or contain a delimiter when matched against a sequence. In our experiments, we use periods as delimiters, but the algorithm allows for any delimiters to be used.

In natural languages, there are many n-gram phrases that act as single units, with meaning distinct from that of each individual word contained in the phrase. Creating phrase patterns with such n-gram phrases (and, therefore, allowing gaps between their words) may alter their meanings. Because the gap size restriction mentioned above applies globally to all phrase patterns, we introduce a flag to disable gaps in certain positions. When an itemset is flagged, no gap is permitted between it and the preceding itemset. To implement this process, in addition to checking for condition (a) and (b) in the scanning process of algorithm 2, we also scan for extension items that can be appended to the prefix without any gap – condition (c). The item is marked differently and its containing itemset is subsequently flagged.

D. Word Classes

Choosing the proper word classes is crucial in dealing with sparse data. There are many options to generate word classes; we explore a mixture of data-driven and linguistic or human knowledge-driven methods, as described below.

Domain-dependent word classes can be learned automatically from unlabeled text data using word clustering algorithms, such as the one introduced by Brown et al. [27]. This algorithm was designed to estimate class-based language models, and the word clustering objective function is the log likelihood of the training data computed using a first-order Markov model. In our experiments, we use the implementation of the algorithm in SRILM [28] to generate two sets of data-driven word classes derived from two different text sources, broadcast conversation (BC) and Wikipedia discussions (introduced below), which will be denoted by *BC auto word classes* and *Wikipedia auto word classes*, respectively.

Linguistic knowledge-driven word classes are constructed based on linguistic knowledge. For example, POS tags can be used as word classes. In this paper, we use the MXPOST English tagger [29] to generate POS word classes for the data. These word classes will be denoted by *POS word classes*.

Hand-constructed word classes are manually designed by experts. Pennebaker et al. created a software program named Linguistic Inquiry and Word Count (LIWC) [30], [31], which identifies word classes in text based on a dictionary. The classes were initially designed to capture various psychometric statistics, from basic linguistic categories such as articles, to more subjective classes, such as words indicative of causal thinking or emotion words. The word classes derived from the LIWC dictionary will be denoted by *LIWC word classes*.

Word classes can also be specifically designed for a particular task or application. In our experiments on alignment classification in Wikipedia discussions, we used 32 word lists constructed by linguists who authored the annotation guidelines for our task. Many word lists contain words frequently used in Wikipedia discussions in contexts relevant to the task, such as words indicating agreement or disagreement, swear words, or modals. The average number of words in these word lists is 22. We also used six word lists that are aggregations of related lists, resulting in additional coarser level word classes. Together, these different lists give 38 word classes (with overlapping membership of words), which will be denoted by *heuristic word classes*.

IV. Experiments

In the paper, we evaluate the performance of phrase pattern features in two tasks, speaker role recognition and alignment classification. The classification performance of n-gram features and phrase pattern features is compared. In initial experiments, we found that phrase pattern features alone do not perform well. Thus, we report only the performance of n-gram features alone or in combination with phrase pattern features. We limit the length of n-grams and phrase patterns to three for a fair comparison. We use the maximum entropy classifier implemented in MALLET [32]. We use binary features, with a value of one to indicate the presence of the feature in an instance, and zero to indicate absence. We use L2 regularization, after obtaining slightly inferior results using L1 regularization in our pilot experiments. The threshold θ used in the mutual information criterion for discriminative phrase pattern mining and the optimal maximum gap size are tuned on development sets for each task.

A. Speaker Role Recognition

In this task, we investigate methods that automatically recognize speaker role in talk shows, based on only lexical cues. Five speaker roles are defined: host, guest participant, audience participant, reporting participant, and a blanket category "other". Invited speakers are considered guest participants, and journalists are labeled as reporting participants.

The dataset consists of 48 English talk shows with human transcripts and speaker segmentation. The speakers in each talk show are labeled by trained annotators with one of the five aforementioned roles. The inter-annotator agreement is $\kappa = 0.67$. We treat the problem of speaker role recognition as a turn-level sequence tagging problem, where a turn is defined as a sequence of uninterrupted speech from a particular speaker (ignoring short backchannel utterances). There are 9009 turns in these talk shows overall.

For consistency of results, we add a global speaker constraint, which forces the turns from the same speaker to have the same role label. Incorporating this constraint in a maximum entropy classifier is relatively straightforward. Suppose that $t_i,\ i = 1, 2, \ldots, n$ are all the turns of a particular speaker, and $p(r|t_i)$ is the posterior probability of role r for turn t_i predicted by the maximum entropy model. Then the final speaker roles for these turns are

$$\hat{r} = \underset{r}{\operatorname{argmax}} \prod_{i=1}^{n} p(r|t_i).$$

In preliminary experiments, we have found that maximum entropy models with the global speaker constraint outperform both hidden Markov models and conditional random fields. Although hidden Markov models and conditional random fields model sequential behavior between turns, the global speaker constraint cannot be easily incorporated in these models, resulting in inferior performance compared to the maximum entropy models with the global speaker constraint.

All experiments are carried out using 5-fold cross validation, and the performance is evaluated using turn-level accuracy. The experimental results are listed in table I, and accuracy differences greater than 0.9% are significant with $p \leq 0.05$ (z-test). The addition of word-only phrase pattern features improves over n-gram features. Only the automatically learned BC word classes lead to additional improvement. If we remove the n-gram features and use phrase pattern features only, the performance can decrease by up to 1% absolute.

Features besides n-grams	Word class type	Accuracy
none	none	85.5%
phrase patterns	none	86.9%
phrase patterns	BC auto	**87.3**%
phrase patterns	Wikipedia auto	85.9%
phrase patterns	POS	85.7%
phrase patterns	LIWC	86.3%

TABLE I
SPEAKER ROLE RECOGNITION RESULTS

While not directly comparable, we can contrast our results with those obtained by other researchers on similar tasks. Liu [26] obtained an accuracy of 82.0% on broadcast news data. Garg et al. [15] reported an accuracy of 78% on the AMI meeting corpus. All these genres have a similar number of participants in each conversation and the conversations are of similar length and degree of complexity, so it is not surprising that the results are also similar.

B. Alignment Classification

Bender et al. [33] introduced the Authority and Alignment in Wikipedia Discussions corpus, containing Wikipedia discussions annotated with various post-level social acts. Previous work on detecting social acts in this corpus has focused on the authority aspect, including detection of turn-level forum authority claims [34] and speaker-level classification of participants in terms of their bids for authority [35]. In this paper, we focus on classification of alignment moves, i.e. expressions of targeted support or dissent between participants.

In this data set, there are 102 discussions for training, 54 for development, and 55 for evaluation. Annotation of alignment moves is a difficult task even for human annotators, with post-level inter-annotator agreement of $\kappa = 0.50$, which roughly corresponds to a post-level F-score of 0.6. Because a single post often contains more than one alignment move, in our experiment, we further segment the posts into sentences, using an English automatic sentence segmenter MXTERMINATOR [36]. The annotations from multiple annotators are consolidated and merged into a single annotation as described in [33]. The classification performance is evaluated using sentence-level recall at the precision-recall break-even point (the operating point at which the system precision is equal to recall). The F-score at this operating point is equivalent to the precision and recall. Since some sentences still contain a mixture of support and dissent, we treat alignment classification as a multi-label classification problem. We build two binary classifiers, one for positive alignment and another for negative alignment, and report the combined performance.

In the experiments, we compare our phrase pattern approach to a heuristic method for taking into account negations used in sentiment analysis [3], where the words in a sentence are marked as negated if they are proceeded by an odd number of negation words selected from a predefined list. Phrase patterns that are built on these negation-marked words are also used as an experimental option. The experimental results are shown in table II. The performance of positive alignment is improved by adding phrase patterns with heuristic word classes, and that of negative alignment is increased after including phrase patterns with Wikipedia word classes. Negation markers benefit n-grams but not phrase patterns. The negation phenomena seem to be better captured through phrase patterns with the original words (vs. the negation-augmented vocabulary). Compared to the n-gram baseline, the best positive alignment performance is significant with $p = 0.10$, and the best negative alignment performance is significant with $p = 0.012$ (paired t-test).

Features besides n-grams	Word class type	Positive alignment	Negative alignment
none	none	0.47	0.43
negation markers	none	0.48	0.43
phrase patterns	none	0.49	0.44
phrase ptn. w/ neg. markers	none	0.48	0.43
phrase patterns	BC auto	0.45	0.44
phrase patterns	Wikipedia auto	0.48	**0.46**
phrase patterns	POS	0.45	0.43
phrase patterns	LIWC	0.42	0.41
phrase patterns	heuristic	**0.50**	0.44

TABLE II
ALIGNMENT CLASSIFICATION RESULTS

Previously reported results are based on different genres. Wang et al. [20] report F-scores of 61.7% for agreement and 55.9% for disagreement detection in broadcast conversations. Hahn et al. [18] report an accuracy of 85.5% for the three-way agreement/disagreement classification task on the ICSI meetings corpus. We note that the performance on the meetings data is substantially higher than on Wikipedia or broadcast speech. We conjecture that this is primarily due to the task definition itself. The task definition used for the broadcast speech data is more closely aligned with our own; here, we hypothesize that the difference is primarily due to genre characteristics.

C. Discussion

From the experimental results for both tasks, we observe some improvements by adding phrase patterns as features. The largest improvements are achieved using phrase patterns with word classes induced from in-domain data or knowledge. On the contrary, generic word classes based on POS and LIWC do not help as much. We conjecture that the reason is because such generic word classes generally have large word clusters, either by virtue of the choice of classes (as in the case of POS-based) or for coverage (for LIWC classes). Large word clusters will likely cause a reduction in the discriminative power of the features, by overly smoothing out the features.

V. Conclusions

We have presented an algorithm to learn discriminative phrase patterns with words and word classes for conversation analysis. The mutual information of phrase patterns with respect to the labels is computed and used as the criterion to filter the phrase patterns in a recursive way. We have looked at the use of these phrase patterns in speaker role recognition and alignment classification, and have reported classification performance improvements.

For future work, we will investigate semi-supervised methods to learning discriminative phrase patterns from both labeled and unlabeled data, where large amount of unlabeled data may be useful in learning phrase patterns that reveal typical language usage.

Acknowledgment

This research was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA). All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI or the U.S. Government. We would also like to thank the UW linguists who suggested the word classes used in this paper.

References

[1] T. Joachims, “Text categorization with support vector machines: Learning with many relevant features,” in *LNCS, Proc. ECML*, vol. 1398, 1998, pp. 137–142.

[2] S. Scott and S. Matwin, “Feature engineering for text classification,” in *Proc. ICML*, 1999, pp. 379–388.

[3] B. Pang, L. Lee, and S. Vaithyanathan, “Thumbs up?: Sentiment classification using machine learning techniques,” in *Proc. EMNLP*, 2002, pp. 79–86.

[4] B. Pang and L. Lee, “A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts,” in *Proc. ACL*, 2004, pp. 271–278.

[5] ——, “Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales,” in *Proc. ACL*, 2005, pp. 115–124.

[6] M. Thomas, B. Pang, and L. Lee, “Get out the vote: Determining support or opposition from congressional floor-debate transcripts,” in *Proc. EMNLP*, 2006, pp. 327–335.

[7] J. Wiebe and E. Riloff, “Creating subjective and objective sentence classifiers from unannotated texts,” *Computational Linguistics and Intelligent Text Processing*, vol. 3406, pp. 486–497, 2005.

[8] G. Sun, X. Liu, G. Cong, M. Zhou, Z. Xiong, J. Lee, and C. Y. Lin, “Detecting erroneous sentences using automatically mined sequential patterns,” in *Proc. ACL*, 2007, pp. 81–88.

[9] B. Zhang, B. Hutchinson, W. Wu, and M. Ostendorf, “Extracting phrase patterns with minimum redundancy for unsupervised speaker role classification,” in *Proc. NAACL-HLT*, 2010, pp. 717–720.

[10] O. Tsur, D. Davidov, and A. Rappoport, “ICWSM – a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews,” in *Proc. AAAI*, 2010.

[11] D. Davidov, O. Tsur, and A. Rappoport, “Semi-supervised recognition of sarcastic sentences in twitter and amazon,” in *Proc. CoNLL*, 2010, pp. 107–116.

[12] R. Barzilay, M. Collins, J. Hirschberg, and S. Whittaker, “The rules behind roles: Identifying speaker role in radio broadcasts,” in *Proc. National Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence*. AAAI Press, 2000, pp. 679–684.

[13] B. Hutchinson, B. Zhang, and M. Ostendorf, “Unsupervised broadcast conversation speaker role labeling,” in *Proc. ICASSP*, 2010, pp. 5322–5325.

[14] Y. Liu, “Initial study on automatic identification of speaker role in broadcast news speech,” in *Proc. NAACL-HLT*, 2006, pp. 81–84.

[15] N. P. Garg, S. Favre, H. Salamin, D. H. Tür, and A. Vinciarelli, “Role recognition for meeting participants: an approach based on lexical information and social network analysis,” in *Proc. the 16th ACM international conference on Multimedia*, 2008, pp. 693–696.

[16] W. Wang, S. Yaman, K. Precoda, and C. Richey, “Automatic identification of speaker role and agreement/disagreement in broadcast conversation,” in *Proc. ICASSP*, 2011, pp. 5556–5559.

[17] D. Hillard, M. Ostendorf, and E. Shriberg, “Detection of agreement vs. disagreement in meetings: training with unlabeled data,” in *Proc. NAACL-HLT*, 2003, pp. 34–36.

[18] S. Hahn, R. Ladner, and M. Ostendorf, “Agreement/disagreement classification: Exploiting unlabeled data using contrast classifiers,” in *Proc. NAACL-HLT*, 2006, pp. 53–56.

[19] M. Galley, K. McKeown, J. Hirschberg, and E. Shriberg, “Identifying agreement and disagreement in conversational speech: Use of bayesian networks to model pragmatic dependencies,” in *Proc. ACL*, 2004, pp. 669–675.

[20] W. Wang, S. Yaman, K. Precoda, C. Richey, and G. Raymond, “Detection of agreement and disagreement in broadcast conversations,” in *Proc. ACL-HLT*, 2011, pp. 374–378.

[21] G. Dong and J. Pei, *Sequence Data Mining*. Springer, 2007.

[22] J. Pei, J. Han, B. Mortazavi-asl, H. Pinto, Q. Chen, U. Dayal, and M.-c. Hsu, “PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth,” in *Proc. International Conference on Data Engineering*, 2001, pp. 215–224.

[23] X. Yan, J. Han, and R. Afshar, “CloSpan: Mining closed sequential patterns in large datasets,” in *Proc. SDM*, 2003, pp. 166–177.

[24] X. Ji, J. Bailey, and G. Dong, “Mining minimal distinguishing subsequence patterns with gap constraints,” *Knowledge and Information Systems*, vol. 11, no. 3, pp. 259–286, April 2007.

[25] Y. Yang and J. O. Pedersen, “A comparative study on feature selection in text categorization,” in *Proc. ICML*, 1997, pp. 412–420.

[26] C.-L. Liu, “Some theoretical properties of mutual information for student assessments in intelligent tutoring systems,” *Foundations of Intelligent Systems*, vol. 3488, pp. 77–93, 2005.

[27] P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai, “Class-based n-gram models of natural language,” *Comput. Linguist.*, vol. 18, no. 4, pp. 467–479, Dec. 1992.

[28] A. Stolcke, “SRILM – an extensible language modeling toolkit,” in *Proc. ICSLP*, 2002, pp. 901–904.

[29] A. Ratnaparkhi, “A maximum entropy part-of-speech tagger,” in *Proc. EMNLP*, 1996, pp. 133–141.

[30] J. Pennebaker, M. Francis, and R. Booth, *Linguistic Inquiry and Word Count: LIWC2001*. Erlbaum Publishers, 2001.

[31] J. Pennebaker, C. Chung, M. Ireland, A. Gonzales, and R. Booth, *The development and psychometric properties of LIWC2007*, 2007.

[32] A. K. McCallum, “MALLET: A machine learning for language toolkit,” 2002, http://mallet.cs.umass.edu.

[33] E. M. Bender, J. T. Morgan, M. Oxley, M. Zachry, B. Hutchinson, A. Marin, B. Zhang, and M. Ostendorf, “Annotating social acts: Authority claims and alignment moves in wikipedia talk pages,” in *Proc. Workshop on Language in Social Media*, 2011, pp. 48–57.

[34] A. Marin, B. Zhang, and M. Ostendorf, “Detecting forum authority claims in online discussions,” in *Proc. Workshop on Language in Social Media*, 2011, pp. 48–57.

[35] A. Marin, M. Ostendorf, B. Zhang, J. T. Morgan, M. Oxley, M. Zachry, and E. M. Bender, “Detecting authority bids in online discussions,” in *Proc. SLT*, 2010.

[36] J. C. Reynar and A. Ratnaparkhi, “A maximum entropy approach to identifying sentence boundaries,” in *Proc. the Fifth Conference on Applied Natural Language Processing*, 1997.