Learning Phrase Patterns for Text Classification Using a Knowledge Graph and Unlabeled Data

Alex Marin[*1], *Roman Holenstein*[2], *Ruhi Sarikaya*[2], *Mari Ostendorf*[1]

[1]University of Washington, Seattle, WA 98195
[2]Microsoft Corporation, Redmond, WA 98052

Abstract

This paper explores a novel method for learning phrase pattern features for text classification, employing a mapping of selected words into a knowledge graph and self-training over unlabeled data. Using Support Vector Machine classification, we obtain improvements over lexical and fully-supervised phrase pattern features in domain and intent detection for language understanding, particularly in conjunction with the use of unlabeled data. Our best results are obtained using unlabeled data filtered for both model training and feature learning based on the confidence of the baseline classifiers.

1. Introduction

Text classification is an important natural language processing application with a wide variety of uses. Often, classification tasks are a component of a larger system - topic classifiers improve the performance of a speech recognition or machine translation system by adapting the language model to the relevant domain or task; sentiment classification is used in recommendation systems by online retailers and media providers. Natural language interfaces to computing systems are becoming more prevalent and require performing a number of classification tasks. For instance, interactive media and gaming devices require detecting the domain or domains of interest to a user (e.g. third party applications or games, music, TV or videos) to improve the user's experience and provide relevant content. Given a domain of interest, the system also must detect the user's intended action before it can provide a meaningful response (e.g. launching an application or displaying movie ratings). In such systems, words and phrases which contain information required by the system to take action are often tagged as "slots".

Typical text classification systems consist of two components, a feature extraction component which is responsible for generating features given the body of text of interest, and the classifier, which assigns a class label to the text (usually, a word sequence). Text classification systems often use lexical features, such as n-grams, and lexicon containment features (i.e. binary features, holding positive value if at least one word in the lexicon occurs in the text), with the lexicons either designed by human experts [1, 2] or automatically-generated [3, 4]. Various word cluster features, such as part-of-speech (POS)-based or automatically-learned (e.g. [5]) have also been used.

Lexical and lexicon containment features are by necessity local and do not capture information about the structure of the sentence. While sufficient for simpler tasks, such as topic classification, more complex problems like intent classification or slot-filling for dialog systems often require some long-distance context features. One approach is to use syntactic features, such as dependency parse tuples (e.g. [4]). However, such features require running expensive parsing models during the evaluation phase. Another approach is to hand-craft regular expressions, which require human effort but are cheaper to apply; they can also be used to implement business decisions or system constraints, which could be difficult to model in a statistical system. A method which merges the benefits of these two approaches involves learning phrase patterns, extensions of n-grams allowing gaps between words. Such features have been used successfully in a variety of text processing tasks [6, 7, 8, 9, 10].

In this paper, we propose a novel method of learning phrase patterns. Section 2 describes the core phrase pattern learning algorithm and contrasts it with previous work. The experimental setup and results are described in section 3. We discuss additional experiments aimed at system analysis in section 4 and conclude with suggestions for future work in section 5.

2. Phrase Pattern Learning

Given a word sequence $[w_1^n]$, we define a phrase pattern as a subsequence $[w_{i_1}, \ldots, w_{i_k}]$ with $1 \leq i_1 < \ldots i_k \leq n$. The words w_{i_k} for each pattern are referred to here as the "pivots" of the pattern. The set of all possible phrase patterns is very large for all but the smallest bodies of text; thus, using phrase patterns as features is impossible without initial pruning of the pattern space. Frequency-based algorithms for pattern selection, such as *PrefixSpan* [11] and *CloSpan* [12] prune the search space using unsupervised methods, while the *ConSGapMiner* [13] uses per-class frequency to perform feature selection. Discriminative methods also exist: the *ExtendedPrefixSpan* algorithm [10] uses a Mutual Information (MI) criterion instead of the frequency criterion used by regular *PrefixSpan*, with patterns being pruned only if they are neither *qualified* (i.e. have sufficiently high MI with the class) nor *promising* (i.e. have no extension with sufficiently high MI with the class).

2.1. Knowledge Graph-Driven Pattern Learning

We propose a new method for learning phrase patterns using external information in the form of a knowledge graph (KG). We constrain the search space by initially accumulating word co-occurrence patterns across sentences, ignoring order, then extracting ordered phrase patterns from the unordered subsets. Thus, longer-span dependencies within each sentence are detected without having to search through the complete set of possible patterns within it. Additionally, instead of learning phrase patterns directly from the connection between words in a sentence and the label of that sentence, we decouple the pattern extraction from the sentence labels to allow more effective use of unlabeled or noisy-labeled data. We relax the label space by mapping each sentence to one or more entries in a KG such as

[*] Work done while at Microsoft

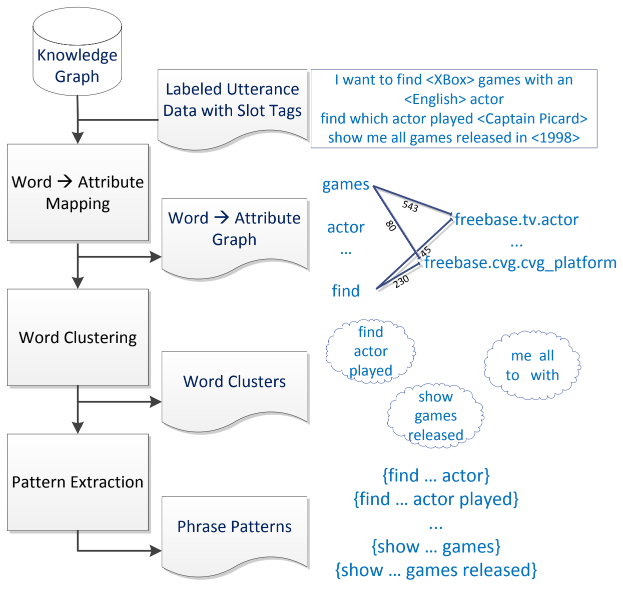

Figure 1: Phrase Pattern Learning

FreeBase [14], and using the "type" attributes of the KG entries (e.g. "freebase.tv.actor" for an actor's name) as intermediate labels. "Type" attributes of KG entries indicate *is-a* relationships, thus providing a natural intermediate label space. The intermediate labels become features in defining word co-occurrence clusters. Processing is split into three stages, as presented (with examples) in figure 1 and described below.

2.1.1. *Word → Attribute Mapping*

We design our mapping of words to attributes (first block in figure 1) using the intuition that KG entries in a given sentence (tagged in "slots") are related to both the domain and task of interest to the user and the desired system action; the remaining words in the sentence provide the "context" regarding the KG entries tagged in each slot. To build our bipartite word → attribute graph, for each word w_i in a given sentence s not belonging to a slot, we define the weight of the edge between w_i and attribute a_j as

$$c_s(w_i, a_j) = \sum_k I(e_{sk}, a_j) \tag{1}$$

where the indicator function I marks whether attribute a_j is associated with the KG entry e_{sk} corresponding to the kth slot of sentence s. Note that the per-sentence weight for a word w_i is not dependent on the word itself, but only on the KG entries present within the sentence s and their attributes a_j. The final edge weight $c(w_i, a_j)$ aggregates the per-sentence counts for all sentences in which w_i appears:

$$c(w_i, a_j) = \sum_s c_s(w_i, a_j) \tag{2}$$

2.1.2. *Pivot Clustering*

We restrict the search space for patterns by clustering the pivots using their KG attribute mappings (middle block in figure 1), using the intuition that words with similar KG attribute distributions will become pivots in the same patterns. We represent each word w_i as a vector of coefficients $c(w_i, a_j)$, normalized by the inverse word frequency of each attribute, analogous to TF-IDF [15] in information retrieval. Pairwise word distances are computed using the cosine distance.

Using these scores, we implement agglomerative clustering [16] with complete linkage to perform the merging of similar clusters. The agglomerative approach gives a natural interpretation to the clusters - each subsequent merge produces a relaxation of the constraints on the pattern search space. The number of clusters used is a tunable parameter of the system.

Since agglomerative clustering is a greedy algorithm, the clustering solution may degenerate into only a small number of large clusters. To avoid this, we add regularization based on the empirical distribution p of the word clusters that penalizes imbalanced $p(c_1), p(c_2)$. The final clustering objective is given in equation 3, where d represents the distance between clusters c_1 and c_2 and λ the tunable regularization weight.

$$(\hat{c}_1, \hat{c}_2) = argmin_{c_1,c_2} d(c_1, c_2) + \lambda \log \frac{p(c_1)p(c_2)}{p(c_1 \cup c_2)} \tag{3}$$

2.1.3. *Ordered Pattern Extraction*

In the final step of the algorithm (last block in figure 1), we extract ordered phrase patterns from unordered pivot clusters. We map the word clusters onto each sentence in the training data, keeping track of sequences of pivots which appear in the same sentence and belong to the same cluster. For each such sequence, we consider all subsequences as possible phrase patterns. Unlike *PrefixSpan* or *ExtendedPrefixSpan*, our pattern extraction algorithm does not require pruning of the patterns during the extraction phase. However, feature selection using frequency or mutual information-based methods may be performed subsequently to reduce the number of model parameters and avoid overtraining. In particular, we filter out the least frequent or degenerate patterns (details in section 3.2). Furthermore, if large amounts of labeled data are available, then the clustering and ordered pattern extraction can be performed separately for each class label. On the other hand, for small datasets, the patterns can be extracted for the entire dataset, ignoring class labels.

2.2. Using Unlabeled Data

Unlabeled data can be used both in model training and for learning more effective features. A common approach for model training in NLP tasks is self-training [17, 18, 19]. To avoid training on noisy labels, at each iteration, only data labeled by the current system with high confidence is added to the training set for the next iteration. We will use this idea in two stages. First, we prune the set of unlabeled data using a simple classifier. Then, we apply self-training with our classifier using phrase pattern features on this subset for model training, with additional methods used for feature learning.

Two filtering techniques are explored for feature learning:

- learn features for long sentences, using the intuition that long sentences are more likely to be mis-classified, and the observation that phrase pattern features are more likely to match (and thus help classify) longer sentences;
- learn features for hard-to-classify sentences, using an empirical definition for classification difficulty: we select those sentences that have low confidence according to a supervised classifier for the same or a related task.

For label-independent feature learning, data for all labels (as previously selected using the simple classifier) are used together.

3. Experimental Results

3.1. Task Overview

We present results for two tasks relevant to the language understanding (LU) component of the Xbox One entertainment and media system: domain classification and intent classification. The domain classification tasks are binary classification tasks, using data from multiple domains in a one-against-all setting. The domains of interest, which act as positive classes for each domain classification task, are “Apps” and “Movies”. The “Apps” domain contains queries related to applications available on the Xbox One; “Movies” consists of queries about movies, actors, and filmography topics. The negative class for domain classification includes data from the other domains used in the Xbox One LU system. Intent classification is domain-dependent; a single, multi-class classification task is set up for each domain. Perfect domain knowledge is assumed in our experiments. There are 8 intents in the “Apps” domain (e.g. *find_app, open_app, show_purchases*). The “Movies” domain has 12 intents (e.g. *find_movie, play_movie, rate_movie*).

Data for all the domains consists of independent, single sentence queries which were collected internally and hand-annotated with domain and intent labels, as well as slot tags relevant to the domain and intent annotations. Each query belongs to a single domain and is annotated with a single intent. Most queries are short (3-7 words), though significant length variation exists within each domain. The annotated data is split 70-10-20 into training, development, and evaluation sets.

To make use of unlabeled data, we collected additional search queries from three sources - text queries from the Bing search desktop and mobile client and transcriptions of spoken web queries. The three sets of queries are used as external data sources for domain and intent classification tasks in the respective domain. The statistics of the hand-labeled and external datasets are presented in table 1, where numbers for the three external sources are based on noisy, automatically-generated labels from the Bing system.

	Domain		
Fileset	Apps	Movies	Other
Train	16k	58k	120k
Dev	2k	7k	15k
Test	4k	14k	30k
Desktop	4k	17k	126k
Mobile	48k	68k	684k
Speech	5k	8k	237k

Table 1: Dataset statistics (# sent.)

3.2. System Design

We evaluate our phrase pattern learning method using a support vector machine (SVM) [20] classifier with a linear kernel. Features in the baseline LU system consist of binary n-gram and lexicon containment features, with the latter defined using hand-crafted (e.g. to represent dates and times) and auto-generated (e.g. the set of all actor names in the KG) lexicons. Phrase pattern features are added to the baseline feature set, since we expect them to be effective primarily in classifying long, complex utterances, for which n-gram features are ineffective. We compare our method for learning phrase patterns with the state-of-the-art discriminative MI-based phrase pattern learning method described in [10], which was shown to outperform frequency-based phrase pattern learning methods. KG patterns for the domain tasks are learned in a label-dependent way. Intent patterns are learned independent of intent labels but using domain-specific data (and thus labels).

Hyperparameter tuning for the phrase pattern learning systems is performed on the development set. For the supervised phrase pattern learning, we tune the maximum number of pivot words in the phrase patterns, as well as the mutual information threshold λ used for pruning. For KG-driven phrase pattern learning, we tune the cluster regularization coefficient and the number of clusters from which phrase patterns are extracted. Additionally, we perform frequency- and length-based pattern filtering, removing degenerate KG patterns (containing a single repeated word) and ngrams which appeared only once in the data. No feature selection was performed beyond the selection of phrase patterns from either method.

3.3. Results Using Labeled Data

Domain	Classification Task	Baseline	MI Pat.	KG Pat.
Apps	Domain	2.9	**2.3**	**2.4**
Apps	Intent	3.6	3.6	3.8
Movies	Domain	12.4	**12.1**	**12.2**
Movies	Intent	4.9	**4.1**	**4.3**

Table 2: Labeled data results, dev (% classification error)

Dev set results on systems trained using the hand-annotated data are presented in table 2 (bold indicates statistically significant improvement over the baseline; significance throughout the paper is computed using a one-sided t-test). We compare our new KG-driven pattern learning method (KG Pat.) against the baseline system and the MI-based method (MI Pat.) Both the MI-based patterns and the KG-based patterns improve upon the baseline in three out of four tasks. Overall, the MI-based method slightly outperforms our KG-based method, though the results are always close, and the KG method does not rely on discriminative learning so it is useful for unlabeled data.

3.4. Results Using External Unlabeled Data

Initial experiments with “Movies” domain classification using all external data for both feature learning and model training yielded some performance degradation (classification error: 12.5% vs. 12.4% baseline). Thus, we focus our efforts on data selection methods for phrase pattern learning. We perform a single round of data selection using two different data selection approaches: length-based filtering and confidence-based filtering. Length-based filtering uses only the longest sentences, counting all words in the sentence (*Full*) or words not tagged as slots (*noSlot*). Confidence-based filtering is applied to the subset of data that has been automatically-labeled by the Bing system as corresponding to the target domain. We select data according to the confidence predicted by the supervised classifiers trained for each task (i.e. 1-vs-all domain classification, *DomainBaseConf*, or within-domain intent classification, *IntentBaseConf*), with the low confidence sentences used for pattern learning and the high confidence sentences used for model training. Tuning the data selection parameters (the minimum utterance length and classifier confidence threshold) resulted in approximately 20% of unlabeled data being selected for pattern learning in each configuration. Two additional configurations apply confidence filtering for both phrase pattern learning and model training - we use the least confident 20% of the unlabeled data for feature learning and use the rest for model training.

Data selection results are shown in table 3 (best results in

Selection Criterion	Apps		Movies	
	Domain	Intent	Domain	Intent
Baseline (no filtering)				
noExternal	2.9	**3.6**	12.4	4.9
Length Filter, Feature Learning				
Full	2.3	4.2	11.3	4.5
noSlot	2.4	4.3	11.4	4.6
Confidence Filter, Feature Learning				
DomainBaseConf	2.4	4.2	11.2	4.6
IntentBaseConf	2.3	4.3	11.2	4.6
Confidence Filter, Feature Learning + Model Training				
DomainBaseConf	2.4	4.3	11.7	4.7
IntentBaseConf	**2.2**	4.2	**11.1**	**4.3**

Table 3: Data selection results, dev (% classification error)

bold). All methods for data filtering for phrase pattern learning (all but the last two rows in table 3) improve over the baseline, but no single method is consistently best. When performing data selection for model training as well (last two rows), the results are more consistent - adding domain classification-based filtering for model training hurts performance, particularly in the "Movies" domain tasks, while intent classification-based filtering yields the best data selection result in each task. This configuration outperforms the baseline in three of the four cases; the difference in the last case is not statistically significant.

Selection Target	Apps		Movies	
	Domain	Intent	Domain	Intent
ModelTrain	2.3	4.3	11.5	4.5
PatLearn	2.3	4.3	11.2	4.6
Model+Pat	**2.2**	**4.2**	**11.1**	**4.3**

Table 4: Intent-based confidence selection, dev (% class. error)

We further evaluate the impact of data selection for feature learning vs. model training, focusing on intent-based confidence filtering, which gave us the overall highest improvement. We compare three configurations: using external data for model training, phrase pattern learning, or both. The model training configuration appends the Bing labels-prefiltered external data classified with highest confidence to the fully-labeled training set. The threshold for data selection is re-tuned for this task. The model is trained using baseline features. The phrase pattern learning configuration and the combined configuration use the threshold tuned for phrase pattern learning, with low confidence sentences used for pattern learning and the remaining set used for model training in the combined case (again after prefiltering). Results are shown in table 4 (best results in bold). We find that the results when only model training or only feature learning make use of the unlabeled data are relatively similar. However, adding the external data to both model training and feature learning yields the best results in all tasks.

Features	Apps				Movies			
	Domain		Intent		Domain		Intent	
	Dev	Eval	Dev	Eval	Dev	Eval	Dev	Eval
Baseline	2.9	2.6	**3.6**	**3.1**	12.4	12.3	4.9	5.1
+MI Pat.	2.3	**2.3**	**3.6**	3.2	12.1	11.9	**4.1**	**4.0**
+KG Pat.	**2.2**	**2.3**	3.8	3.5	**11.1**	**11.5**	4.3	4.2

Table 5: Final results, dev and eval (% classification error)

The final results are presented in table 5 (best results in bold). The KG phrase patterns yield the best results in the two domain classification tasks; both results are statistically significant against the baseline. The best KG pattern system is also statistically significant when compared to the best supervised MI pattern system in "Movies" domain classification; both significantly outperform the baseline system for intent classification in the movies domain, with the supervised patterns holding a slight (but statistically-insignificant) edge. In "Apps" intent classification, the baseline system performs as well as or better than all pattern features, though statistically all results are tied.

4. Analysis

Our hyperparameter tuning for KG patterns focused on the two cluster parameters: number of clusters and regularization coefficient. We find that the classification results are fairly robust with respect to both parameters, with $< 1\%$ classification error increase when tuning the regularization coefficient over the range of $0.00001 - 0.01$, and over a large range of cluster counts. On the other hand, the number of patterns learned varies quite significantly with the number of clusters used during the pattern learning process. Since no additional feature selection is performed, adding a large number of patterns to the feature set, relative to the initial number of features, may lead to a higher risk of overtraining. Thus, selecting hyperparameters which yield a small number of pattern features is preferred.

To evaluate the impact of our data selection approach, as an alternative to performing data selection for pattern learning, we use the entire set of Bing labels-filtered training sentences, but weight each sentence $\rightarrow$ attribute connection by the baseline classification confidence of that sentence. We examine the performance of this alternative method in the movies domain and intent classification tasks. For domain classification, we obtain a small, statistically insignificant (0.1% accuracy) degradation from the weighting method compared to filtering for feature learning. The performance remains unchanged in the intent classification task. Thus, confidence weighting may be an acceptable alternative to filtering for some applications.

Comparing the complexity of the two phrase pattern learning methods, the mutual information-based method requires $O(e^{|L|})$ operations, where $|L|$ is the number of class labels (here, $|L| < 15$). In the knowledge graph-based method, the clustering requires $O(|V|^3)$ operations, where $|V|$ is the number of non-KG entry words in the vocabulary (here, $|V| < 5000$), while the pattern extraction is exponential in the maximum number of pivots in a pattern (which is a fixed, but tunable parameter). The mapping of sentences to KG attributes requires $O(|V|)$ operations, assuming the number of KG type attributes is fixed and small (relative to $|V|$) and the KG is structured efficiently as a distributed database.

5. Conclusions

We have introduced a novel method for learning phrase pattern features for text classification tasks, using a mapping of entities present in text into an external knowledge graph. Such features are competitive with those learned using state-of-the-art discriminative methods, and outperform the supervised-only features when used in a semi-supervised setting.

Variations on this work that may lead to improved performance include multi-domain and multi-task word-attribute graphs for use in word clustering and pattern extraction, and combining MI-based patterns learned on labeled data with KG-based patterns that leverage external resources. Additional data selection approaches for both feature learning and model training can also be studied.

6. References

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

[2] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *Proceedings of HLT-EMNLP*, Stroudsburg, PA, USA, 2005, pp. 347–354.

[3] Alex Marin, Mari Ostendorf, Bin Zhang, Jonathan T. Morgan, Meghan Oxley, Mark Zachry, and Emily M. Bender, "Detecting authority bids in online discussions," in *Proceedings of SLT*, 2010, pp. 49–54.

[4] Alex Marin, Bin Zhang, and Mari Ostendorf, "Detecting forum authority claims in online discussions," in *Proceedings of Workshop on Language in Social Media*, 2011, pp. 48–57.

[5] Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai, "Class-based n-gram models of natural language," *Comput. Linguist.*, vol. 18, no. 4, Dec. 1992, pp. 467–479.

[6] S. Jaillet, A. Laurent, and M. Teisseire, "Sequential patterns for text categorization," *Intell. Data Anal.*, vol. 10, no. 3, 2006, pp. 199–214.

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

[8] Guihua Sun, Xiaohua Liu, Gao Cong, Ming Zhou, Zhongyang Xiong, John Lee, and Chin Y. Lin, "Detecting erroneous sentences using automatically mined sequential patterns," in *Proceedings of ACL*, 2007, pp. 81–88.

[9] Oren Tsur, Dmitry Davidov, and Ari Rappoport, "ICWSM – a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews," in *Proceedings of AAAI*, 2010.

[10] Bin Zhang, Alex Marin, Brian Hutchinson, and Mari Ostendorf, "Learning phrase patterns for text classification," *IEEE Transactions on Audio Speech and Language Processing*, , no. 6, 2013, pp. 1180–1189.

[11] Jian Pei, Jiawei Han, Behzad Mortazavi-asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-chun Hsu, "PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *Proceedings of International Conference on Data Engineering*, 2001, pp. 215–224.

[12] Xifeng Yan, Jiawei Han, and Ramin Afshar, "CloSpan: Mining closed sequential patterns in large datasets," in *Proceedings of SDM*, 2003, pp. 166–177.

[13] Xiaonan Ji, J. Bailey, and Guozhu Dong, "Mining minimal distinguishing subsequence patterns with gap constraints," in *Proceedings of International Conference on Data Mining*, 2005, pp. 194–201.

[14] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of SIGMOD*, 2008, pp. 1247–1250.

[15] Karen Spärck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, 1972, pp. 11–21.

[16] D. Wishart, "Mode analysis: a generalization of nearest neighbor which reduces chaining effects," in *Numerical Taxonomy*, A. J. Cole, Ed. 1969, pp. 282–308, Academic Press, London and New York.

[17] David McClosky, Eugene Charniak, and Mark Johnson, "Effective Self-Training for Parsing," in *Proceedings of the HLT-NAACL*, 2006.

[18] Shan He and Daniel Gildea, "Self-training and co-training for semantic role labeling," Tech. Rep. 891, University of Rochester, 2006.

[19] Bin Wang, Bruce Spencer, CharlesX. Ling, and Harry Zhang, "Semi-supervised self-training for sentence subjectivity classification," in *Advances in Artificial Intelligence*, Sabine Bergler, Ed., vol. 5032 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008, pp. 344–355.

[20] Corinna Cortes and Vladimir Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, Sept. 1995, pp. 273–297.