Hard vs. Fuzzy Clustering for Speech Utterance Categorization

Amparo Albalate[1] and David Suendermann[2]

[1] Institute of Information Technology, University of Ulm
amparo.albalate@uni-ulm.de,
[2] SpeechCycle Inc. NY, USA,
david@speechcycle.com

Abstract. To detect and describe categories in a given set of utterances without supervision, one may apply clustering to a space therein representing the utterances as vectors. This paper compares hard and fuzzy word clustering approaches applied to 'almost' unsupervised utterance categorization for a technical support dialog system. Here, 'almost' means that only one sample utterance is given per category to allow for objectively evaluating the performance of the clustering techniques. For this purpose, categorization accuracy of the respective techniques are measured against a manually annotated test corpus of more than 3000 utterances.

1 Introduction

A technical support automated agent is a spoken language dialog system devised to perform problem solving tasks over the phone in a similar way as human agents do [Acomb et al., 2007]. These systems are nowadays adopted as an efficient solution to common problems in technical support call centers, such as the long waiting time experienced by the user, the cost of training and maintaining a large base of human agents, and the scalability of the service.

One of the main features of technical support automated agents is the natural modality of interaction with the callers. With an open-ended prompt (e.g. *Please briefly describe the reason for your call*), the system leaves the interaction initiative to the users who are allowed to provide a description of the problem with their own words. Because a symptom can be spoken in multiple ways and styles (natural language), user utterances in response to open prompts may become very complex. A robust semantic analysis of the input utterance is thus necessary in order to identify the underlying problem or symptom from the description provided by the caller. Once the problem has been diagnosed, the automated agent can guide the caller through the relevant troubleshooting steps towards the problem solution.

In technical support applications, the semantic analysis of natural language utterances is commonly based on a technology called statistical spoken language understanding (SSLU). In essence, SSLU performs a mapping of user utterances into one of the predefined problem categories. This is generally achieved through statistical pattern recognition based on supervised classifiers which are trained with manually labelled utterance sets in order to automatically classify new unlabeled utterances.

However, the manual compilation of large training corpora requires an extensive human labeling effort with high associated time cost: when data for a new application is collected, a number of sample utterances is manually analyzed and initial categories are defined. Then, a large set of utterances is labelled according to these categories. In doing so, the human labeler potentially faces difficulties assigning particular utterances to the predefined set of categories. The labeler will then iteratively extend and alter the category set to cover also utterances he has problems with. On the other end, it is completely unclear if the category set and its definition are reasonable from the statistical classification point of view. If category definitions are too vague, labelers tend to label similar (and sometimes even identical) utterances with different categories. It is also possible that, due to the feature set it relies on, the classifier cannot distinguish between utterances whereas the human labeler can. E.g., the utterances

> *schedule appointment* (the caller wants to set up an appointment) and
> *appointment schedule* (the caller is calling about his appointment which potentially has been setup earlier)

would be labelled with different categories for their meaning being different. A unigram classifier (ignoring the order in which words appear) is unable to distinguish between these utterances, though.

All these issues can potentially be overcome by means of unsupervised categorization methods which aim at producing category definitions by themselves optimizing the separability of the categories. This is done by taking all (non-labelled) utterances of a given training set into account which then are implicitly labelled while optimizing the category definitions.

Unfortunately, it is very hard to analyse the quality of the categories, an unsupervised algorithm comes up with, without a lot of human involvement (basically human labelers going over the algorithm's suggestions and subjectively rating them). Therefore, in this publication, we trigger a basically unsupervised categorization algorithm by means of a single manual example per category providing suggestions on the number and very gross locations of the reference categories. Then, we use a manually labelled test set to estimate the algorithm's performance. As opposed to the above mentioned procedure for evaluating a completely unsupervised technique, this test is repeatable, cheap and allows for frequent tuning cycles.

How can, however, a single example sufficiently represent the entire diversity of utterances in a category? What about the fact that there can be several ways to express the same (or similar) meanings? In the domain this work is carried out (automated troubleshooting for cable television), two example caller utterances for the category *NoPicture* are

– *no picture,*
– *no image,*

and two examples for the category *NoSound* are

– *no sound,*
– *no audio*.

A straight-forward clustering algorithm being given, say, the first of the utterance pairs as the single mentioned training example, would not have the words *image* and *audio* in its vocabulary. It, hence, would assign both *no image* and *no sound* with the same likelihood to the categories *NoPicture* and *NoSound*. If the algorithm knew that *picture* and *image* are synonyms, as *sound* and *audio* are, it would assign the utterances correctly.

Therefore, in this paper, we discuss feature extraction methods which aim at capturing semantic relationships between words such as synonymy and polysemy. In particular, we analyse and compare two approaches to the classification of words based on hard and fuzzy clustering.

The remainder of the paper is organized as follows: In Section 2, we present an overview of the utterance categorization modules. In the next section, we pay special attention to the feature extraction module incorporating either hard or fuzzy clustering. Finally, we describe our evaluation methods and discuss results as well as further directions of the work in Sections 4 and 5, respectively.

2 Utterance categorization with three modules

Automated speech utterance categorization was introduced about ten years ago to allow the caller to use unconstraint natural speech to express the call reason [Gorin et al., 1997]. At the same time, speech utterance categorization was capable of distinguishing many more reasons than directed dialogs, common at that time, could ever handle.

There is a number of approaches to statistical speech utterance categorization (see [Evanini et al., 2007]) which, however, are based on a significant amount of manually labelled training data. Being provided only a single training utterance per category requires special modifications of the categorization procedure as discussed in the following.

Figure 1 provides an overview of the utterance categorization model which consists of three sequential modules: preprocessing, feature extraction, and categorization.

2.1 Preprocessing

The preprocessing module applies morphological analysis, stop word filtering, and bag-of-word representation.

First, a Part of Speech (POS) tagger [Toutanova and Manning, 2000] and a morphological analyser [Minnen et al., 2001] have been applied to reduce the surface word forms in utterances into their corresponding lemmas.

As a next step, stop words are eliminated from the lemmas, as they are judged irrelevant for the categorization. Examples are the lemmas *a, the, be, for*. In this work, we used the SMART stop word list [Buckley, 1985] with small modifications: in particular, we deleted confirmation terms (*yes* and *no*) from the list, whereas words typical for spontaneous speech (*eh, ehm, uh*) were treated as stop words.

The categoriser's vocabulary is then defined as the set of distinct lemmas in the preprocessed training utterances: $W = (w_1, \ldots, w_D)$. In this work, the vocabulary dimension is $D = 1614$ lemmas.

Finally, the lemmas for each utterance are combined as a *bag of word*. I.e., each utterance is represented by a D-dimensional vector whose binary elements represent

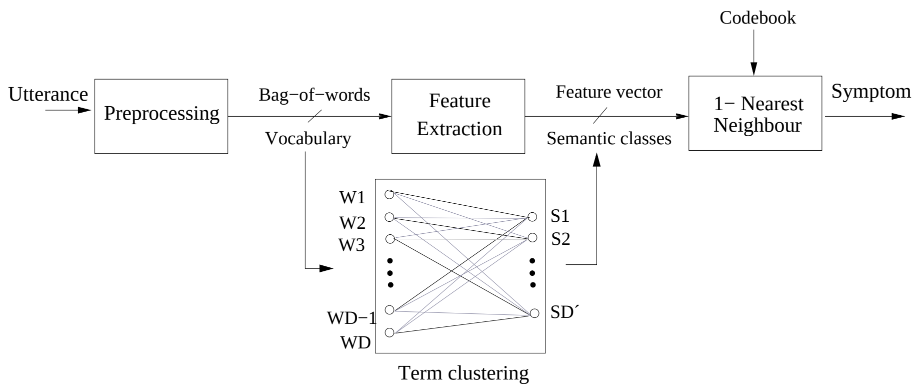

Fig. 1. Utterance categorization components. For feature extraction, hard and fuzzy approaches to term clustering are compared. Hard clustering can be seen as a hard mapping of each input pattern into a single output class (black traces). In contrast, a fuzzy clustering provides a *fuzzy* or *soft* association of each pattern to the output classes through a membership matrix (grey lines). Hard clustering can also be observed as a particular case of fuzzy clustering, where pattern memberships are either ’1’ or ’0’.

the presence/absence of the respective vocabulary element in the current utterance:

$$BW = (b_1, \ldots, b_D) \tag{1}$$

2.2 Feature extraction

In order to extract the set of salient features for utterance categorization, we apply clustering to the vocabulary terms. The rational for the use of term[3] clustering is the need for extracting semantic effects, such as synonymy and polysemy, which may not be represented in the original bag of words. Here, we distinguish between hard and fuzzy term clustering.

In hard term clustering, each input pattern is unequivocally allocated to one output cluster. This approach may be adequate for capturing semantically related terms (e.g. synonyms) in output semantic classes. In contrast, a fuzzy clustering algorithm associates the input patterns to all output classes through a matrix with membership degrees. If a considerable number of polysemous terms (with several related meanings) is present in the input data, fuzzy techniques should then be more appropriate.

After the feature extraction phase, each input bag of words (BW) is accordingly transformed into a feature vector F. Details of the feature extraction based on hard and fuzzy clustering and the representation of new feature vectors are discussed in Sections 3.2 and 3.3, respectively.

[3] In the following, we also use *term* as a synonym for *lemma*.

2.3 Utterance categorization

In order to categorise a test utterance represented by its bag of words or feature vector into one of N categories, we use the Nearest Neighbor (NN) algorithm. This algorithm requires a codebook of prototypes composed of one labelled utterance per category. Each input utterance is then assigned to the category of the closest prototype. The proximity of an input utterance to the prototypes is here calculated according to the inner product between their feature vectors, F_a and F_b:

$$s(F_a, F_b) = F_a \cdot F_b. \tag{2}$$

3 Term clustering

3.1 Term vector of lexical co-occurrences

A frequently reported problem to word clustering is the adequate representation of word lemmas in vector structures so that mathematical (dis)similarity metrics applied to term vectors can reflect the terms' semantic relationships [Montgomery, 1975]. We follow a second-order term co-occurrence criterion [Picard, 1999] for detecting word-semantic proximities:

Two words are similar to the degree that they co-occur with similar words.

Consequently, each vocabulary term w_i is represented in a D-dimensional vector of lexical co-occurrences:

$$W_i = (c_{i1}, \ldots, c_{iD}) \tag{3}$$

wherein the constituents c_{ij} denote the co-occurrence of the terms w_i and w_j, normalized with respect to the total sum of lexical co-occurrences for the term w_i:

$$c_{ij} = \frac{nc_{ij}}{\sum\limits_{k \neq i} nc_{ik}}. \tag{4}$$

Here, nc_{ij} denotes the total number of times that w_i and w_j co-occur. Finally, in order to extract the terms' semantic dissimilarities, we have used the Euclidean distance between term vectors.

3.2 Hard term clustering

As mentioned in Section 2.2, a hard clustering algorithm places each input pattern into a single output cluster. Based on the complete-link criterion [Johnson, 1967], the proposed term clustering[4] produces a partition of the vocabulary terms given an input user parameter, the maximum intra-cluster distance d_{th}:

[4] The proposed clustering algorithm is a variant of the complete link which uses the cluster merging condition from complete link, while the search criterion is based on a single link approach. Thus, this procedure meets the complete link condition for the maximum intra cluster distance, and simultaneously prevents relatively close patterns to be assigned into different clusters. Note, however, that no hierarchical structure (dendogram) can be drawn for the output partitions. A hierarchical alternative would be a *centroid* or *average link* approach.

1. Construct a dissimilarity matrix U between all pairs of patterns. Initially, each pattern composes its individual cluster $c_k = \{w_k\}$.
2. Find the patterns w_i and w_j with minimum distance in the dissimilarity matrix.
   - If the patterns found belong to different clusters, $c_a \neq c_b$, and $U_{max}(c_a, c_b) \leq d_{th}$, where $U_{max}(c_a, c_b)$ is the distance of the furthest elements in c_a and c_j, merge clusters c_a and c_b.
   - Update U so that $U_{ij} = \infty$.
3. Repeat step 2) while $U_{min} \leq d_{th}$ or until all patterns remain assigned to a single cluster.

As a result of the hard term clustering algorithm, different partitions of the vocabulary terms are obtained, depending on the input parameter d_{th}. Because the elements in each cluster should indicate terms with a certain semantic affinity, we also denote the obtained clusters as *semantic classes*. Table 1 shows examples of clusters produced by this algorithm.

Table 1. Example utterances of semantic classes obtained by hard term clustering for $d_{th1} = d$ on a text corpus comprising 30,000 running words from the cable television troubleshooting domain; the average number of terms per cluster is 4.71; the total number of extracted features is 1458.

speak, talk
operator, human, tech, technical, customer, representative, agent, somebody, someone, person, support, service
firewall, antivirus, protection, virus, security, suite, program, software, cd, driver
reschedule, confirm, cancel, schedule
remember, forget
webpage, site, website, page, web, message, error, server
megabyte, meg
technician, appointment
update, load, download
boot, shut, turn
user, name, login, usb
area, room, day

After hard term clustering, the bag of words remains represented in a binary feature vector F_{hard}:

$$F_{hard} = (b_{f_1}, b_{f_2}, \ldots, b_{f_{D'}}) \tag{5}$$

where the b_{f_i} component denotes the existence of at least one member of the i^{th} extracted class in the original bag of words.

Disambiguation. If applied to bags of words or feature vectors extracted from hard term clusters, the NN classifier rejects a considerable number of ambiguous utterances for which several candidate prototypes are found[5]. A disambiguation module has been therefore developed to resolve the mentioned ambiguities and map an ambiguous utterance to one of the output categories.

First, utterance vectors with more than one candidate prototype are extracted. For each pattern, we have a list of pointers to all candidate prototypes. Then, the terms in each pattern that cause the ambiguity are identified and stored in a competing term list.

As an example, let us consider the utterance *I want to get the virus off my computer* which, after pre-processing and hard term clustering, results in the feature set *computer get off virus*. Its feature vector has maximum similarity to the prototypes *computer freeze* (category *CrashFrozenComputer*) and *install protection virus* (category *Security*). The competing terms that produce the ambiguity are in this case the words *computer* and *virus*. Therefore, the disambiguation among prototypes (or categories) is here equivalent to a disambiguation among competing terms. For that reason, as a further means of disambiguation, we estimate the *informativeness* of a term w_i as shown in Equation 6:

$$I(w_i) = -(log(Pr(w_i)) + \alpha \cdot log(\sum_{\substack{j \\ L_j = N}} c_{ij} Pr(w_j))) \tag{6}$$

where $Pr(w_i)$ denotes the maximum-likelihood estimation for the probability of the term w_i in the training corpus, and L_j refers to the part-of-speech (POS) tag of w_j, where N refers to nouns). POS tags have been extracted by means of the Standford POS tagger [Toutanova and Manning, 2000].

As it can be inferred from Equation 6, two main factors are taken into account in order to estimate the relevance of a word for the disambiguation:

a) the word probability and
b) the terms' co-occurrence with frequent nouns in the corpus.

The underlying assumption that justifies this second factor is that words representative of problem categories are mostly nouns and appear in the corpus with moderate frequencies. The parameter α is to control the trade-off between the two factors. Reasonable values are in the range of $(\alpha \in [1, 2])$ placing emphasis on the co-occurence term; for our corpus, we use $\alpha = 1.6$ which we found best-performing in the current scenario.

Finally, the term with highest informativeness is selected among the competitors, and the ambiguous utterance vector is matched to the corresponding prototype or category.

3.3 Fuzzy term clustering

The objective of the fuzzy word clustering used for feature extraction is a fuzzy mapping of words into semantic classes and leads to the membership matrix M repre-

[5] Candidate prototypes are such prototypes which share maximum proximity to the input utterance. This happens specially when the similarity metric between the vectors results in integer values, e.g. in the case of using the inner product of binary vectors as the aforeintroduced bags of words and feature vectors extracted after hard word clustering.

senting this association. We use the Pole-based overlapping clustering (PoBOC) algorithm [Cleuziou et al., 2004] which distinguishes two kinds of patterns: poles and residuals.

Poles are homogeneous clusters which are as far as possible from each other. In contrast, residuals are outlier patterns that fall into regions between two or more poles. The elements in the poles represent monosemous terms, whereas the residual patterns can be seen as terms with multiple related meanings (polysemous). The PoBOC algorithm is performed in two phases: (i) pole construction, and (ii) multiaffectation of outliers.

In the **pole construction** stage, the set of poles $\{P\} = \{P_1, \cdots, P_{D'}\}$ and outliers $\{R\}$ are identified and separated. Poles arise from certain terms, known as the *pole generators*, with a maximal separation inside a dissimilarity graph.

In the **multi-affectation** stage, the outliers' memberships to each pole in $\{P\}$ are computed. Finally, the term w_i is assigned a membership vector to each P_j pole as follows:

$$M_{ij} = \begin{cases} 1, & \text{if } w_i \in P_j \\ 1 - d_{av}(W_i, P_j)/d_{max} & \text{if } w_i \in \{R\} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where $d_{av}(w_i, P_j)$ denotes the average distance of the w_i word to all objects in P_j, and d_{max} is the maximum of the term dissimilarity matrix.

Finally, the feature vector obtained with fuzzy term clustering, F_{fuzzy}, is calculated as the normalized matrix product between the original bag of words BW and the membership matrix M:

$$F_{fuzzy} = \frac{BW_{(1xD)} \cdot M_{(DxD')}}{|BW \cdot M|}. \tag{8}$$

4 Experiments

In order to evaluate the proposed hard and fuzzy word clustering methods for utterance classification, we compare the performance of an NN classifier directly applied to the bag of word vectors with that after performing feature extraction. As introduced in Section 1, this is done by comparing the output categories the proposed algorithm assigns to a number of test utterances with manually assigned categories thereof (the reference). If both categories coincide, the automatic categorization is considered correct, otherwise it is counted as error. As overall accuracy, we define

$$\text{accuracy} = \frac{\text{\# correctly classified test utterances}}{\text{\# total utterances in test set}} \tag{9}$$

In the following, we describe the test corpus on which we evaluated the proposed algorithms. Then, we report on the experimental results and finally discuss the outcomes.

4.1 Corpus description

We used a corpus of 34,848 transcribed and annotated caller utterances gathered from user interactions of a commercial video troubleshooting agent. From this corpus, 31,535 utterances were used for training[6] and 3,285 utterances for test. The remaining 28 utterances (one labelled utterance per category) were manually selected as NN prototypes. Most of the original utterances are composed of 1 to 10 words. After preprocessing, we have an average of 4.45 terms per utterance. The final vocabulary is composed of $D = 1614$ terms; we distinguish $N = 28$ distinct categories in this work.

4.2 Results

Table 2 shows accuracies on the test set achieved by several configurations of the NN classifier: (i) no feature extraction (bag-of-word matching), (ii) fuzzy term clustering, and (iii) best partition obtained with hard word clustering. Results obtained with the further use of the disambiguation procedure directly applied to bags of words or feature vectors after hard word clustering are also presented. Finally, as a standard of comparison, we also report the accuracy of a 'trivial' classifier which assigns the most frequent category to every utterance.

Table 2. Results of utterance categorization experiments using several feature extraction techniques.

Classifier	Clustering	Disambiguation	Accuracy
trivial	–	–	12.5%
NN	–	no	45.0%
NN	–	yes	57.0%
NN	Fuzzy	no	50.0%
NN	Hard	no	50.8%
NN	Hard	yes	62.2%

A second comparison of the utterance classification accuracy rates obtained with hard and fuzzy clustering methods is shown in Figure 2. In the case of hard clustering, results make reference to different cluster partitions obtained with distinct values of the intra cluster threshold distance (d_{th}), normalized with respect to the largest value ($d_{th_{max}}$) used in our experiments[7]. Also, for hard clustering, results are provided before and after disambiguation.

4.3 Discussion

Looking at the plain results without the use of disambiguation, it turns out that both hard and fuzzy word clustering achieve almost the same accuracy (around 50%) out-

[6] As training corpus we refer to the utterances used in the feature extraction module for lexical analysis and term clustering. None of these methods makes use of the utterances' manual annotations.

[7] The range of d_{th} values used in hard clustering has been selected through an analysis of the term distances histogram, so that a majority of words are enclosed in this range.

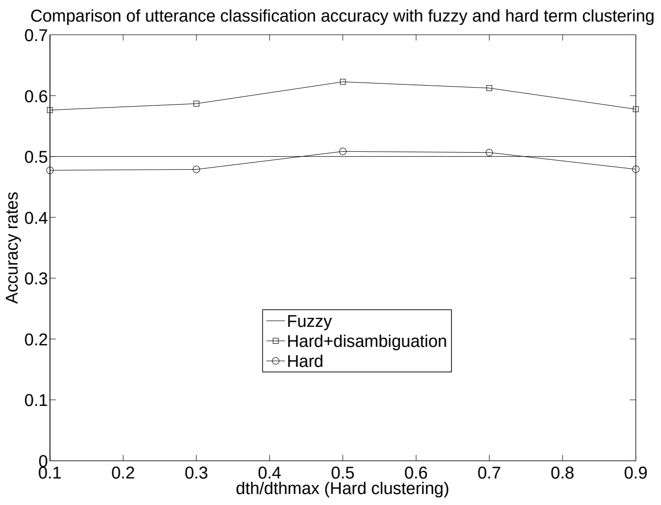

Fig. 2. Comparison of utterance categorization accuracies obtained with feature extraction based on fuzzy and hard clustering, before and after the disambiguation procedure (hard clustering case) is applied.

performing the baseline NN classifier that uses raw bag-of-word vectors by more than 5%.

Although both performances are similar, the nature of the classification errors made by fuzzy and hard word clustering potentially differ. In the fuzzy clustering case, feature vectors are composed of real numbers whereas the hard clustering vectors are binary (cf. Section 3.2). Hence, the distance measures in the former case are real, in the latter case integer. As for the example in Section 3.2, it often happens that one or more competing categories result in the very same distance leading to a considerable number of ambiguous cases.

This fact motivated the use of the disambiguation module. Indeed, the disambiguation led to significant improvements in the utterance categorization performance. The accuracy maximum (62.2%) is reached by the combination of hard term clustering and disambiguation. This classifier configuration outperforms the NN classifier with disambiguation by 5.2% and the baseline by a considerable 17.2%.

5 Conclusion

Given only one sample utterance per category, the proposed categorization scheme produces up to 62.2% correct classification results in our test scenario using hard term clustering as feature extraction in conjunction with a disambiguation procedure. Without disambiguation, utterance categorization accuracies by up to 50% are reached by both fuzzy and hard term clustering. The classification errors observed with hard word clustering are partially due to ambiguities produced during feature vector matching. In this latter case, the categorization can potentially benefit from the further use of a disambiguation scheme as demonstrated in our experiments. We can thus conclude that the most appropriate utterance categorization scheme among the analyzed techniques is based on hard term clustering.

In the future we aim at studying bootstrapping techniques which help enlarge very small training sets automatically. Also subject of analysis are new utterance (dis)similarity metrics which make direct use of terms' semantic (dis)similarities, in order to avoid the limited performance of an intermediate term clustering approach. This is to increase the ratio between correct and incorrect categorizations being one of the most important criteria in commercially deployed applications.

References

[Acomb et al., 2007] Acomb, K., Bloom, J., Dayanidhi, K., Hunter, P., Krogh, P., Levin, E., and Pieraccini, R. (2007). Technical Support Dialog Systems: Issues, Problems, and Solutions. In *Proc. of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, Rochester, USA.

[Buckley, 1985] Buckley, C. (1985). Implementation of the SMART information retrieval system. Technical report, Cornell University, Ithaca, USA.

[Cleuziou et al., 2004] Cleuziou, G., Martin, L., and Vrain, C. (2004). PoBOC: An Overlapping Clustering Algorithm. Application to Rule-Based Classication and Textual Data. In *Proc. of the ECAI*, Valencia, Spain.

[Evanini et al., 2007] Evanini, K., Suendermann, D., and Pieraccini, R. (2007). Call Classification for Automated Troubleshooting on Large Corpora. In *Proc. of the ASRU*, Kyoto, Japan.

[Gorin et al., 1997] Gorin, A., Riccardi, G., and Wright, J. (1997). How May I Help You? *Speech Communication*, 23(1/2).

[Johnson, 1967] Johnson, S. (1967). Hierarchical Clustering Schemes. *Psychometrika*, 32.

[Minnen et al., 2001] Minnen, G., Carrol, J., and Pearce, D. (2001). Applied Morphological Processing of English. *Natural Language Engineering*, 7(3).

[Montgomery, 1975] Montgomery, C. A. (1975). A Vector Space Model for Automatic Indexing. *Communication of the ACM*, 18(11).

[Picard, 1999] Picard, J. (1999). Finding Content-Bearing Terms using Term Similarities. In *Proc. of the EACL'99*, Bergen, Norway.

[Toutanova and Manning, 2000] Toutanova, K. and Manning, C. (2000). Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In *Proc. of the EMNLP/VLC*, Hong Kong, China.