A Self-Learning Universal Concept Spotter

Tomek Strzalkowski and Jin Wang
GE Corporate Research and Development
P.O. Box 8
Schenectady, NY 12301
USA
{strzalkowski,wangj}@crd.ge.com

Abstract

We describe the Universal Spotter, a system for identifying in-text references to entities of an arbitrary, user-specified type, such as people, organizations, equipment, products, materials, etc. Starting with some initial seed examples, and a training text corpus, the system generates rules that will find further concepts of the same type. The initial seed information is provided by the user in the form of a typical lexical context in which the entities to be spotted occur, e.g., "the name ends with *Co.*", or "to the right of *produced* or *made*", and so forth, or by simply supplying examples of the concept itself, e.g., *Ford Taurus, gas turbine, Big Mac.* In addition, negative examples can be supplied, if known. Given a sufficiently large training corpus, an unsupervised learning process is initiated in which the system will: (1) find instances of the sought-after concept using the seed-context information while maximizing recall and precision; (2) find additional contexts in which these entities occur; and (3) expand the initial seed-context with selected new contexts to find even more entities. Preliminary results of creating spotters for organizations and products are discussed.

1 Introduction

Identifying concepts in natural language text is an important information extraction task. Depending upon the current information needs one may be interested in finding all references to people, locations, dates, organizations, companies, products, equipment, and so on. These concepts, along with their classification, can be used to index any given text for search or categorization purposes, to generate summaries, or to populate database records. However, automating the process of concept identification in unformatted text has not been an easy task. Various single-purpose spotters have been developed for specific types of concepts, including people names, company names, location names, dates, etc. but those were usually either hand crafted for particular applications or domains, or were heavily relying on apriori lexical clues, such as keywords (e.g., 'Co.'), case (e.g., 'John K. Big'), predicatable format (e.g., 123 Maple Street), or a combination of thereof. This makes creation and extension of such spotters an arduous manual job. Other, less salient entities, such as products, equipment, foodstuff, or generic references of any kind (e.g., 'a Japanese automaker') could only be identified if a sufficiently detailed domain model was available. Domain-model driven extraction was used in ARPA-sponsored Message Understanding Conferences (MUC); a detailed overview of current research can be found in the proceedings of MUC-5 (muc5, 1993) and the recently concluded MUC-6, as well as Tipster Project meetings, or ARPA's Human Language Technology workshops (tipster1, 1993), (hltw, 1994).

We take a somewhat different approach to identify various types of text entities, both generic and specific, without a detailed understanding of the text domain, and relying instead on a combination of shallow linguistic processing (to identify candidate lexical entities), statistical knowledge acquisition, unsupervised learning techniques, and possibly broad (universal but often shallow) knowledge sources, such as on-line dictionaries (e.g., WordNet, Comlex, OALD, etc.). Our method moves beyond the traditional name spotters and towards a universal spotter where the requirements on what to spot can be specified as input parameters, and a specific-purpose spotter could be generated automatically. In this paper, we describe a method of creating spotters for entities of a specified category given only initial seed examples, and using an unsupervised learning process to discover rules for finding more instances of the concept. At this time we place no limit on what kind of things one may want to build a spotter for, although our experiments thus far concentrated on entities customarily re-

ferred to with noun phrases, e.g., equipment (e.g., "gas turbine assembly"), tools (e.g., "adjustable wrench"), products (e.g., "canned soup", "Arm & Hammer baking soda"), organizations (e.g., American Medical Association), locations (e.g., Albany County Airport), people (e.g., Bill Clinton), and so on. We view the semantic categorization problem as a case of disambiguation, where for each lexical entity considered (words, phrases, N-grams), a binary decision has to be made whether or not it is an instance of the semantic type we are interested in. The problem of semantic tagging is thus reduced to the problem of partitioning the space of lexical entities into those that are used in the desired sense, and those that are not. We should note here that it is acceptable for homonym entities to have different classification depending upon the context in which they are used. Just as the word "bank" can be assigned different senses in different contexts, so can "Boeing 777 jet" be once a product, and another time an equipment and not a product, depending upon the context. Other entities may be less context dependent (e.g., company names) if their definitions are based on internal context (e.g., "ends with *Co.*") as opposed to external context (e.g., "followed by *manufactures*"), or if they lack negative contexts.

The user provides the initial information (seed) about what kind of things he wishes to identify in text. This information should be in a form of a typical lexical context in which the entities to be spotted occur, e.g., "the name ends with *Co.*", or "to the right of *produced* or *made*", or "to the right of *maker of*", and so forth, or simply by listing or highlighting a number of examples in text. In addition, negative examples can be given, if known, to eliminate certain 'obvious' exceptions, e.g., "not to the right of *made for*", "not *toothbrushes*". Given a sufficiently large training corpus, an unsupervised learning process is initiated in which the system will: (1) generate initial context rules from the seed examples; (2) find further instances of the sought-after concept using the initial context while maximizing recall and precision; (3) find additional contexts in which these entities occur; and (4) expand the current context rules based on selected new contexts to find even more entities.

In the rest of the paper we discuss the specifics of our system. We present and evaluate preliminary results of creating spotters for organizations and products.

2 What do you want to find: seed selection

If we want to identify some things in a stream of text, we first need to learn how to distinguish them from other items. For example, company names are usually capitalized and often end with 'Co.', 'Corp.', 'Inc.' and so forth. Place names, such as cities, are normally capitalized, sometimes are followed by a state abbreviation (as in *Albany, NY*), and may be preceded by locative prepositions (e.g., *in, at, from, to*). Products may have no distinctive lexical appearance, but they tend to be associated with verbs such as 'produce', 'manufacture', 'make', 'sell', etc., which in turn may involve a company name. Other concepts, such as equipment or materials, have few if any obvious associations with the surrounding text, and one may prefer just to point them out directly to the learning program. There are texts, e.g., technical manuals, where such specialized entities occur more often than elsewhere, and it may be advantagous to use these texts to derive spotters.

The seed can be obtained either by hand tagging some text or using a naive spotter that has high precision but presumably low recall. A naive spotter may contain simple contextual rules such as those mentioned above, e.g., for organizations: a noun phrases ending with "Co." or "Inc."; for products: a noun phrase following "manufacturer of", "producer of", or "retailer of". When such naive spotter is difficult to come by, one may resort to hand tagging.

3 From seeds to spotters

The seed should identify the sought-after entities with a high precision (though not necessarily 100%), however its recall is assumed to be low, or else we would already have a good spotter. Our task is now to increase the recall while maintaining (or even increase if possible) the precision.

We proceed by examining the lexical context in which the seed entities occur. In the simplest instance of this process we consider a context to consist of N words to the left of the seed and N words to the right of the seed, as well as the words in the seed itself. Each piece of significant contextual evidence is then weighted against its distribution in the balance of the training corpus. This in turn leads to selection of some contexts to serve as indicators of relevant entities, in other words, they become the initial rules of the emerging spotter.

As an example, let's consider building a spotter for company names, starting with seeds as illustrated in the following fragments (with seed contexts highlighted):

> ... HENRY KAUFMAN is president of *Henry Kaufman &* **Co.**, a ... Gabelli, chairman of *Gabelli Funds* **Inc.**; Claude N. Rosenberg ... is named president of Skandinaviska Enskilda Banken ... become vice chairman of the state-owned electronics giant Thomson S.A. ... banking group, said the formal merger of Skanska Banken into ... water maker Source Perrier S.A., according to French stock ...

Having "Co." "Inc." to pick out "Henry Kaufman & Co." and "Gabelli Funds Inc." as seeds, we proceed to find new evidence in the training corpus, using an unsupervised learning process, and discover that "chairman of" and "president of" are very likely to precede company names. We expand our initial set of rules, which allows us to spot more companies:

> ... HENRY KAUFMAN is **president of** *Henry Kaufman &* **Co.**, a ... Gabelli, **chairman of** *Gabelli Funds* **Inc.**; Claude N. Rosenberg ... is named **president of** *Skandinaviska Enskilda Banken* ... become vice **chairman of** *the state-owned electronics giant Thomson S.A.* ... banking group, said the formal merger of Skanska Banken into ... water maker Source Perrier S.A., according to French stock ...

This evidence discovery can be repeated in a bootstrapping process by replacing the initial set of seeds with the new set of entities obtained from the last iteration. In the above example, we now have "Skandinaviska Enskilda Banken" and "the state-owned electronics giant Thomson S.A." in addition to the initial two names. A further iteration may add "S.A." and "Banken" to the set of contextual rules, and so forth. In general, entities can be both added and deleted from the evolving set of examples, depending on how exactly the evidence is weighted and combined. The details are explained in the following sections.

4 Text preparation

In most cases the text needs to be preprocessed to isolate basic lexical tokens (words, abbreviations, symbols, annotations, etc), and structural units (sections, paragraphs, sentences) whenever applicable. In addition, part-of-speech tagging is usually desirable, in which case the tagger may need to be re-trained on a text sample to optimize its performance (Brill, 1993), (Meteer, Schwartz & Weischedel, 1991). Finally, a limited amount of lexical normalization, or stemming, may be performed.

The entities we are looking for may be expressed by certain types of phrases. For example, people names are usually sequences of proper nouns, while equipment names are contained within noun phrases, e.g., 'forward looking infrared radar'. We use part of speech information to delineate those sequences of lexical tokens that are likely to contain 'our' entities. From then on we restrict any further processing on these sequences, and their contexts.

These preparatory steps are desirable since they reduce the amount of noise through which the learning process needs to plow, but they are not, strictly speaking, necessary. Further experiments are required to determine the level of preprocessing required to optimize the performance of the Universal Spotter.

5 Evidence items

The semantic categorization problem described here displays some parallels to the word sense disambiguation problem where homonym words need to be assigned to one of several possible senses, (Yarowsky, 1995), (Gale, Church & Yarowsky, 1992), (Brown, Pietra, Pietra & Mercer, 1991). There are two important differences, however. First, in the semantic categorization problem, there is at least one open-ended category serving as a grab bag for all things non-relevant. This category may be hard, if not impossible, to describe by any finite set of rules. Second, unlike the word sense disambiguation where the items to be classified are known apriori, we attempt to accomplish two things at the same time:

1. discover the items to be considered for categorization;
2. actually decide if an item belongs to a given category, or falls outside of it.

The categorization of a lexical token as belonging to a given semantic class is based upon the information provided by the words occurring in the token itself, as well as the words that precede and follow it in text. In addition, positional relationships among these words may be of importance. To capture this information, we define the notion of an *evidence set* for a lexical unit $W_1 W_2 ... W_m$ (a phrase, or an N-gram) as follows. Let $...W_{-n}...W_{-1}W_1...W_mW_{+1}W_{+2}...W_{+n}...$ be a string of subsequent tokens (e.g., words) in text, such that $W_1 W_2 ... W_m$ is a unit of interest (e.g., a noun phrase) and n is the maximum size of the context window on either side of the unit. The actual window size may be limited by boundaries of structural units such as sentences or paragraphs. For each unit $W_1 W_2 ... W_m$, a set of *evidence items* is collected as a set union of the following four sets:

1. Pairs of $(word, position)$, where $position \in \{\mathbf{p}, \mathbf{s}, \mathbf{f}\}$ indicates whether *word* is found in the context preceding (**p**) the central unit, following (**f**) it, or whether it comes from the central unit itself (**s**). $E_1 =$

$$\left\{ \begin{array}{cccc} (W_{-n}, \mathbf{p}) & ..... & (W_{-2}, \mathbf{p}) & (W_{-1}, \mathbf{p}) \\ (W_1, \mathbf{s}), & (W_2, \mathbf{s}) & ..... & (W_m, \mathbf{s}) \\ (W_{+1}, \mathbf{f}), & (W_{+2}, \mathbf{f}) & ..... & (W_{+n}, \mathbf{f}) \end{array} \right\}$$

2. Pairs of *(bi-gram, position)* to capture word sequence information. $E_2 =$

$$\left\{ \begin{array}{ccc} ((W_{-n}, W_{-(n-1)}), \mathbf{p}) & ... & ((W_{-2}, W_{-1}), \mathbf{p}) \\ ((W_1, W_2), \mathbf{s}) & ... & ((W_{m-1}, W_m), \mathbf{s}) \\ ((W_{+1}, W_{+2}), \mathbf{f}) & ... & ((W_{+(n-1)}, W_{+n}), \mathbf{f}) \end{array} \right\}$$

3. 3-tuples *(word, position, distance)*, where *distance* indicates how far *word* is located relative to W_1 or W_m. $E_3 =$

$$\left\{ \begin{array}{ccc} (W_{-n}, \mathbf{p}, n) & ... & (W_{-1}, \mathbf{p}, 1) \\ (W_1, \mathbf{s}, m) & ... & (W_m, \mathbf{s}, 1) \\ (W_{+1}, \mathbf{f}, 1) & ... & (W_{+n}, \mathbf{f}, n) \end{array} \right\}$$

4. 3-tuples *(bi-gram, position, distance)*.$E_4 =$

$$\left\{ \begin{array}{c} ((W_{-n}, W_{-(n-1)}), \mathbf{p}, n-1)...(W_{-2}, W_{-1}), \mathbf{p}, 1) \\ ((W_1, W_2), \mathbf{s}, m-1).....((W_{m-1}, W_m), \mathbf{s}, 1) \\ ((W_{+1}, W_{+2}), \mathbf{f}, 1)...((W_{+(n-1)}, W_{+n}), \mathbf{f}, n-1) \end{array} \right\}$$

For example, in the fragment below, the central phrase *the door* has the context window of size 2:

... boys kicked *the door* with rage ...

The set of evidence items generated for this fragment, i.e., $E_1 \cup E_2 \cup E_3 \cup E_4$, contains the following elements:

$$\left\{ \begin{array}{c} (boys, \mathbf{p}), (kicked, \mathbf{p}), (the, \mathbf{s}), \\ (door, \mathbf{s}), (with, \mathbf{f}), (rage, \mathbf{f}), \\ ((boys, kicked), \mathbf{p}), ((the, door)), \mathbf{s}), \\ ((with, rage), \mathbf{f}), (boys, \mathbf{p}, 2), \\ (kicked, \mathbf{p}, 1), (the, \mathbf{s}, 2), (door, \mathbf{s}, 1), \\ (with, \mathbf{f}, 1), (rage, \mathbf{f}, 2), \\ ((boys, kicked), \mathbf{p}, 1), ((the, door)), \mathbf{s}, 1), \\ ((with, rage), \mathbf{f}, 1) \end{array} \right\}$$

Items in evidence sets are assigned *significance weights* (SW) to indicate how strongly they point towards or against the hyphothesis that the central unit belongs to the semantic category of interest to the spotter. The significance weights are acquired through corpus-based training.

6 Training

Evidence items for all candidate phrases in the training corpus, for those selected by the initial used-supplied seed, as well as for those added by a training iteration, are divided into two groups. Group A items are collected from the candidate phrases that are accepted by the spotter; group R items come from the candidate phrases that are rejected. Note that A and R may contain repeated elements.

For each evidence item t, its significance weight is computed as:

$$SW(t) = \left\{ \begin{array}{ll} \frac{f(t,A)-f(t,R)}{f(t,A)+f(t,R)} & f(t,A)+f(t,R) > s \\ 0 & \text{otherwise} \end{array} \right. \quad (1)$$

where $f(t, X)$ is the frequency of t in group X, and s is a constant used to filter the noise of very low frequency items.

As defined $SW(t)$ takes values from -1 to 1 interval. $SW(t)$ close to 1.0 means that t appears nearly exclusively with the candidates that have been accepted by the spotter, and thus provides the strongest positive evidence. Conversely, $SW(t)$ close to -1.0 means that t is a strong negative indicator since it occurs nearly always with the rejected candidates. $SW(t)$ close to 0 indicates neutral evidence, which is of little or no consequence to the spotter. In general, we take $SW(t) > \epsilon > 0$ as a piece of positive evidence, and $SW(t) < -\epsilon$ as a piece of negative evidence, as provided by item t. Weights of evidence items within an evidence set are then combined to arrive at the compound context weight which is used to accept or reject candidate phrase.

At this time, we make no claim as to whether (1) is an optimal formula for calculating evidence weights. An alternative method we considered was to estimate certain conditional probabilities, similarly to the formula used in (Yarowsky, 1995):

$$SW(t) = \log \frac{P(p \in A/t)}{P(p \in R/t)} \approx \log \frac{f(t,A)f(A)}{f(t,R)f(R)} \quad (2)$$

Here $f(A)$ is (an estimate of) the probability that any given candidate phrase will be accepted by the spotter, and f(R) is the probability that this phrase is rejected, i.e., $f(R) = 1 - f(A)$. Thus far our experiments show that (1) produces better results than (2). We continue investigating other weighting schemes as well.

7 Combining evidence weights to classify phrases

In order to classify a candidate phrase, all evidence items need to be collected from its context and their SW weights are combined. When the combined weight exceeds a threshold value, the candidate is accepted and the phrase becomes available for tagging by the spotter. Otherwise, the candidate is rejected, although it may be reevaluated in a future iteration.

There are many ways to combine evidence weights. In our experiments we tried the following two options:

$$x \oplus y \equiv \left\{ \begin{array}{ll} x + y - xy & \text{if } x > 0 \text{ and } y > 0 \\ x + y + xy & \text{if } x < 0 \text{ and } y < 0 \\ x + y & \text{otherwise} \end{array} \right. \quad (3)$$

and

$$x \oplus y \equiv \left\{ \begin{array}{ll} x & \text{if } abs(x) > abs(y) \\ y & \text{otherwise} \end{array} \right. \quad (4)$$

In (3), $x \oplus y$ is greater than either x or y when both x and y are positive, and it is less than both x and y for negative x and y. In all cases, $x \oplus y$ remains within $[-1, +1]$ interval.

In (4) only the dominating evidence is considered. This formula is more noise resistant than (3), but produces generally less recall.

8 Bootstrapping

The evidence training and candidate selection cycle forms a bootstrapping process, as follows:

Procedure Bootstrapping
Collect seeds
loop
Training phase
Tagging phase
until Satisfied.

The bootstrapping process allows for collecting more and new contextual evidence and increase recall of the spotter. This is possible thanks to overall redundancy and repetitiveness of information, particularly local context information, in large bodies of text. For example, in our three-sectional context representation (preceding, self, following), if one section contains strong evidence that the candidate phrase is selectable, evidence found in other sections will be considered in the next training cycle, in order to select additional candidates.

An important consideration here is to maintain an overall precision level throughout the entire process. Although, it may be possible to recover from some misclassification errors (e.g., (Yarowsky, 1995)), care should be taken when adjusting the process parameters so that precision does not deteriorate too rapidly. For instance, acceptance thresholds of evidence weights, initially set high, can be gradually decreased to allow more recall while keeping precision at a reasonable level.

In addition, (Yarowsky, 1995), (Gale, Church & Yarowsky, 1992) point out that there is a strong tendency for words to occur in one sense within any given discourse ("one sense per discourse"). The same seems to apply to concept selection, that is, multiple occurrences of a candidate phrase within a discourse should all be either accepted or rejected by the spotter. This in turn allows for bootstrapping process to gather more contextual evidence more quickly, and thus to converge faster producing better results.

9 Experiments and Results

We used the Universal Spotter to find *organizations* and *products* in a 7 MBytes corpus consisting of articles from the Wall Street Journal. First, we pre-processed the text with a part-of-speech tagger and identified all simple noun groups to be used as candidate phrases. 10 articles were set aside and hand tagged as key for evaluation. Subsequently, seeds were constructed manually in form of contextual rules. For organizations, these initial rules had a 98% precision and 49% recall; for products, the corresponding numbers were 97% and 42%. (4) is used to combine evidences. No lexicon verification (see later) has been used in order to show more clearly the behavior the learning method itself (the performance can be enhanced by lexicon verification). Also note that the quality of the seeds affects the performance of the final spotter since they define what type of concept the system is supposed to look for. The seeds that we used in our experiments are quite simple, perhaps too simple. Better seeds may be needed (possibly developed through an interaction with the user) to obtain strong results for some categories of concepts.

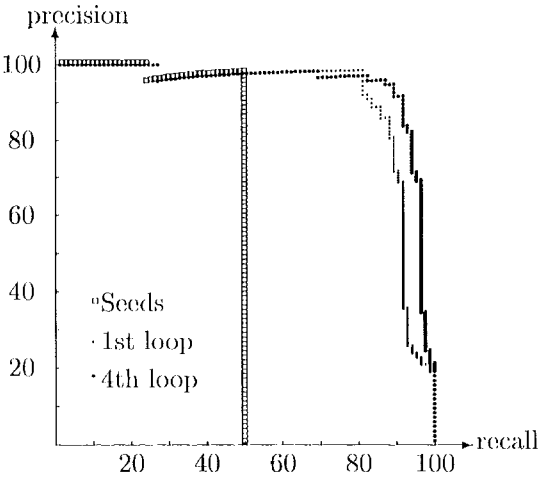

Figure 1: Organization spotter results.

For organization tagging, the recall and precision results obtained after the first and the fourth bootstrapping cycle are given in Figure 1.

The point with the maximum precision*recall in the fourth run is 95% precision and 90% recall. Examples of extracted organizations include: *"the State Statistical Institute Istat", "Wertheim Schroder & Co", "Skandinaviska Enskilda Banken", "Statistics Canada".*

The results for products tagging are given in Figure 2 on the next page. Examples of extracted products include: *"the Mercury Grand Marquis and Ford Crown Victoria cars", "Chevrolet Prizm", "Pump shoe", "AS/400".*

The effect of bootstrapping is clearly visible in both charts: it improves the recall while maintaining or even improving the precision. We may also notice that some misclassifications due to an imperfect seed (e.g., see the first dip in precision on the products chart) can in fact be corrected in further bootstrapping loops. The generally lower performance levels for the product spotter is probably due to the fact that the concept of product is harder to circumscribe.

10 Further options

10.1 Lexicon verification

The items identified in the second step can be further validated for their broad semantic classification using on-line lexical databases such as Com-

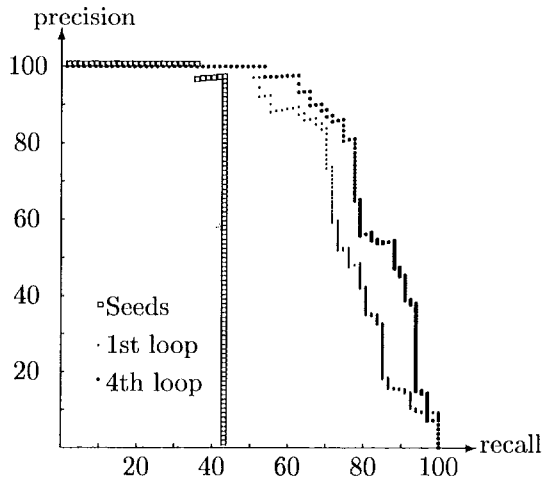

Figure 2: Product spotter results.

lex or Longman Dictionary, or Princeton's WordNet; (Miller, 1990) For example, "gas turbine" is an acceptable equipment/machinery name since 'turbine' is listed as "machine" or "device" in WordNet hierarchy. More complex validation may involve other words in the phrase (e.g., "circuit breaker") or words in the immediate context.

10.2 Conjunctions

The current program cannot deal with conjunction. The difficulty with conjunction is not with classification of the conjoined noun phrases (it is easier, as a matter of fact, because they carry more evidences) but with identification of the phrase itself because of the structural ambiguities it typically involves that cannot be dealt with easily on lexical or even syntactic level.

11 Conclusions

In this paper we presented the Universal Spotter, a system that learns to spot in-text references to instances of a given semantic class: people, organizations, products, equipment, tools, to name just a few. A specific class spotter is created through an unsupervised learning process on a text corpus given only an initial user-supplied seed: either a number of examples of the concept, or a typical context in which they can be found. The experiment shows that this method indeed can produce useful spotters based on easy-to-construct seeds. The results shown here are promising, can be further improved by using lexicon verification. Different methods of computing SWs, combining SWs, and parameter adjustmenting for the bootstrapping process need to be explored as we believe there is still room for improvement. The method is being continuously refined as we gain more feedback from empirical tests across several different applications.

We believe that the Universal Spotter can replace much of the need to create hand-crafted concept spotters commonly used in text extraction operations. In can also be applied to building other than the most common spotters such as those for people names, place names, or company names. In fact, is can be used to create more-or-less on-demand spotters, depending upon the applications and its subject domain. In particular, we believe such spotters will be required to gain further advance in intelligent text indexing and retrieval applications, text summarization, and database applications, e.g., (Harman, 1995), (Strzalkowski, 1995).

References

hltw. 1994. *Proceedings of the Human Language Technology Workshop*, Princeton. San Francisco, CA:Morgan Kaufman Publishers.

muc5. 1993. *Proceedings of 5th Message Understanding Conference*, Baltimore. San Francisco, CA:Morgan Kaufman Publishers.

tipster1. 1993. *Tipster Text Phase 1: 24 month Conference*, Fredericksburg, Virginia.

Brill, E. 1992. A Simple Rule-based Part of Speech Tagger. *Proceedings of 3rd Applied Natural Language Processing* , San Francisco, CA:Morgan Kaufman Publishers.

Brown,P., S. Pietra, V. Pietra and R. Mercer. 1991. Word Sense Disambiguation Using Statistical Methods. *Proceedings of the 29h Annual Meeting of the Association for Computational Linguistics*, pp. 264-270.

Gale, W., K. Church and D. Yarowsky. 1992. A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities*, 26, pp. 415-439.

Harman, D. 1995. Overview of the Third Text REtrieval Conference. *Overview of the Third Text REtrieval Conference (TREC-3)*, pp.1-20.

Meteer, M., R. Schwartz, and R. Weischedel. 1991. Studies in Part of Speech Labeling. *Proceedings of the 4th DARPA Speech and Natural Language Workshop*, Morgan-Kaufman, San Mateo, CA. pp. 331-336.

Miller, G. 1990. WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3, 4.

Strzalkowski, T. 1995. Natural Language Information Retrieval. *Information Processing and Management, vol. 31, no. 3*, pp. 397-417.

Yarowsky, D. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189-196.