Classifying Unknown Proper Noun Phrases Without Context

Joseph SMARR
Symbolic Systems Program
Stanford University
Stanford, CA 94305-2181
jsmarr@stanford.edu

Christopher D. MANNING
Computer Science Department
Stanford University
Stanford, CA 94305-9040
manning@cs.stanford.edu

Abstract

We present a probabilistic generative model used to classify unknown Proper Noun Phrases into semantic categories. The core of the classifier is an n-gram character model, which is enhanced with an n-gram word-length model and a common word model. While most work has depended largely on context or domain-specific rules for semantic disambiguation of unknown names, we demonstrate that there is surprisingly reliable statistical information available in the composition of the names themselves. Using the context-independent probabilities assigned by our domain independent classifier is sufficient to achieve greater than 90% classification accuracy on typical tasks.

Keywords: *named-entity classification, unknown words, probabilistic modeling, n-grams*

Introduction

Unknown words and phrases are a continual source of trouble for statistical NLP techniques, because no statistics can be generated for an item that you have never seen before (Mikheev, 1997). This problem is particularly acute in domains where there are large numbers of specific names, or where new names are constantly being created. For this study, we looked at five such categories of Proper Noun Phrases (PNPs): drug names, company names, movie titles, place names, and people's names.

There has been a great deal of interest in the *named entity extraction* task of extracting PNPs from text and classifying them. For example, the MUC tests require identifying PNPs and classifying them as company names, place names, and people names (Mikheev et al, 1998, Collins et al, 1999). Similarly, several studies have tried to identify novel terms in medical abstracts (e.g. Campbell, et al, 1999, Bodenreider et al, 2000). Here we focus solely on the latter task of classifying PNPs.

Traditional approaches to PNP classification rely primarily on large, manually constructed lists of known names or "gazetteers" (Wacholder et al, 1997, Charoenpornsawat et al, 1998, Mikheev et al, 1998, Bodenreider et al, 2000), hand-built, domain-specific rules based on patterns in the syntactic context (Appelt et al, 1995, Mikheev et al, 1998, Bodenreider et al, 2000), and/or gross word-level features such as capitalization, punctuation, or presence of numbers and other "special characters" (Bikel et al 1997, Wacholder et al, 1997, Baluja et al, 1999, Bikel et al, 1999, Bodenreider et al, 2000). Many of these systems use some form of machine learning in conjunction with these features. These methods achieve relatively high levels of performance, but suffer from the problem that building lists of words or heuristic rules is slow, expensive, and domain-specific. These shortcomings have been widely acknowledged (e.g., Wacholder et al, 1997, Mihkeev et al, 1999), yet few alternative strategies for unknown PNP classification have been proposed.

In this paper we show that the internal composition of PNPs provides surprisingly strong evidence for classification, independent of any context. People often find it easy to classify PNPs that they have never seen before: a PNP like *Novo-Doxylin* just looks like a drug name. In an application of named entity classification, the context of use of a PNP often provides powerful clues to classification, and these are exploited by most systems. Our central

contention is that current systems insufficiently exploit information in the word shape of a PNP (i.e., essentially phonaesthetic considerations), which can also be effectively used for classification. By word shape, we mean features like common letter sequences and word lengths, along with the presence of key words within the PNP (*Inc., Jr.*, etc.).

In a complete system, our classifier would serve as an informed *prior probability* over PNP classes, which could be combined with additional clues from context, for instance using Bayesian evidence combination. However, we focus here solely on the problem of computing context-independent priors, which has so far been under explored.

The value of word-level features has been exploited in part-of-speech tagging, where suffix morphology is used to decide the class of unknown words (e.g., Mikheev, 1997). This technique would not work for semantic classification, but we propose a method of building a probabilistic model of the generation of PNPs, which can exploit latent regularities in word sequencing and shape. The method is similar to the use of character n-grams for language identification (Dunning 1994, Cavnar and Trenkle 1994), and essentially reuses known techniques, though tuned to the features of the case at hand.

The contribution of this paper is thus not so much in the novelty of the methods used, but in showing how much value can be gotten from an information source that has been ignored. Moreover, by automatically learning the statistics of a given category of PNPs, rather than manually constructing domain-specific heuristics, we also realize the ability to quickly reach high levels of classification accuracy in novel domains, a challenge that few existing methods have met.

1 Formalization of Problem

1.1 Task

The performance task is to take a string representing a Proper Noun Phrase (e.g. *Aureomycin* or *Keyspan Corp*) and classify it into one of a predefined set of categories (e.g. *drug name* or *company name*). A *Proper Noun Phrase* is a sequence of one or more words that represents the name of a person, place, or thing. As mentioned above, our goal is to assess the ability to classify an already segmented PNP independent of context. Segmentation of PNPs from text is a separate and prior problem, which has been well studied (Abney, 1991, Ramshaw et al, 1995, Bikel et al, 1997).

1.2 Training

We use a standard supervised learning paradigm. A training example consists of a PNP labeled with its semantic category. A portion of the training examples (20% in the reported results) are held-out for learning various parameters described in the next section, and the remainder are counted to derive various statistics. After cross-validation parameters have been set, the held-out data is also trained on before testing.

1.3 Testing

After training is completed, the classifier is presented with another list of PNPs, none of which appeared in the training data. Some PNPs are inherently ambiguous and thus could be judged as correctly belonging to multiple classes (e.g., *Washington* is both a place and a name). However, we follow the stringent evaluation standard of only accepting the category from which the example was originally taken, regardless of its ambiguity.

1.4 Evaluation

Each result presented is the average of 10 separate train/test runs. In each case, a randomly selected 90% of the supervised data is used for training, and the remaining 10% is used for testing. The evaluation metric we use is raw classification accuracy, defined as the number of test examples for which the correct category was provided by the classifier, divided by the total number of test examples presented.

2 Model Used for Classification

Classification of PNPs is performed using a probabilistic generative model for each category. Classification is determined as follows:

$$\text{Predicted-Category}(pnp) = \text{argmax}_c\ P(c|pnp)$$
$$= \text{argmax}_c\ P(c)^{\alpha} \times P(pnp|c)$$

P(c) is estimated empirically from the training data, and α is a prior-boost set with a line search

on held-out data.[1] P(*pnp*|*c*) is a model of each category of PNPs, which is based principally on two types of features: the number and length of words used, and the composition of each word.

Formally, P(*pnp*|*c*) is a generative model of the following form (each term is implicitly conditioned on the category):

$$P(pnp|c) = P_{\text{n-gram}}(\textbf{word-lengths}(pnp)) \times \prod_{\text{word } i \in pnp} P(w_i|\textbf{word-length}(w_i))$$

$$P(w_i|len) = \lambda_{len} \times P_{\text{n-gram}}(w_i|len)^{k/len} + (1-\lambda_{len}) \times P_{\text{word}}(w_i|len)$$

The probability for each word is computed as the weighted average of the character n-gram estimate and the known-word estimate. Interpolation weights are learned for each distinct word length using a line search on held-out data.[2] P(*w*|*len*) is estimated empirically as the number of times word *w* was seen in the training data divided by the total number of words of length *len* characters seen.

The function **word-lengths**(*pnp*) returns a list of integers, representing the number of characters in each word of the PNP. The list is prepended with (n–1) boundary symbols for conditioning, and a final "0-lenth word" indicating the end of the PNP is also added so that the termination of PNPs becomes a statistically observable event. For example, **word-lengths**(*Curtis E. Lemay*) = [6, 2, 5, 0].

Two n-gram (Markov) models are used, one for word lengths, and one for characters. We set *n*=4 for lengths, and *n*=6 for characters, and used deleted interpolation to estimate probabilities, using the following recursive function:

$$P_{\text{0-gram}}(symbol|history) = \text{uniform-distribution}$$

$$P_{\text{n-gram}}(s|h) = \lambda_{C(h)} P_{\text{empirical}}(s|h) + (1 - \lambda_{C(h)}) P_{\text{(n-1)-gram}}(s|h)$$

Thus, the 2-gram estimate is a mix of the empirical 2-gram distribution and the combined 1/0-gram distribution, the 3-gram estimate is a mix of the empirical 3-gram and the combined 2/1/0-gram, etc. Interpolation parameters $\lambda_{C(h)}$ are estimated via EM on held-out data, and are learned separately based on the binned count of the conditioning context C(*h*). We fixed the following bins in our experiments: {0, ≤5, ≤50, ≤500, ≤5000, >5000}, though we found only minor deviation in our results by changing the specific bins used.

The character n-gram estimate for the entire word is conditioned on word-length by dividing by the fraction of words in the training data with the given number of characters.[3] The first word in the PNP is prepended with n-1 spaces (starter symbols, with the effect that the 2-gram and lower estimates treat the first word identically to a middle word) and the subsequent words are prepended with the preceding n-1 chars in the PNP (including the preceding space). Each estimate is run up through the following space, and for the last word, a unique termination symbol is appended.

Working with character level models means that longer words have more influence on classification than short words. To mitigate the effects of this bias, we normalize the n-gram estimate for length by taking the (*k*/*length*)'th root, where *k* is a global constant learned on held-out data through a line search. The motivation for this process and its results are discussed in more detail in Section 5.3.2.

3 Data Sets Used in Experiments

We assembled five categories of PNPs for our experiments, each containing several thousand examples (see the appendix for a complete information on counts and sources). The categories were pharmaceutical drugs (*drug*), companies listed on the New York Stock Exchange (*nyse*), movies produced in 2000 (*movie*), cities and countries from around the world (*place*), and famous people's names (*person*). These collections were selected because they represented major sources of unknown PNPs of interest, and because of their diverse composition.

These data sets were intentionally left in the rather "noisy" state in which they were found, to breed robustness, and to accurately measure performance on "real world" examples. There is in-

[1] Setting a prior boost usually made little difference as it was often set to 1.0 during cross-validation.

[2] If a novel word length is seen during testing, the interpolation parameter is automatically set to 1 for the character n-gram and 0 for the word model, since no words of that length have been seen before.

[3] We condition the probability for each word on its length in order to keep it independent from the probability of seeing any word of that length, which is already computed by the length n-gram model.

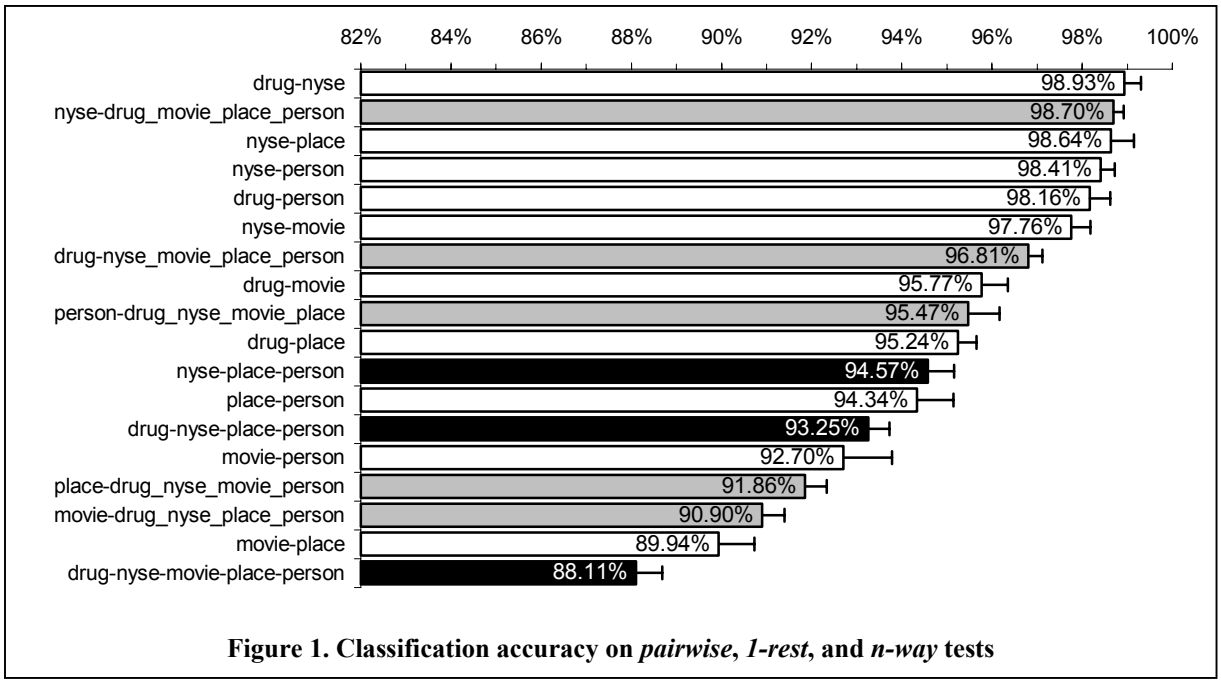

Figure 1. Classification accuracy on *pairwise*, *1-rest*, and *n-way* tests

consistent use of capitalization, punctuation, and canonical formatting. Many of the PNPs within a given category come from different languages (e.g., foreign films). Some categories contain a number of frequently occurring words (e.g., *Inc.* and *Corporation* in *nyse*); others do not.[4]

It has been pointed out in the MUC competitions that PNPs often appear abbreviated in text, especially when mentioned earlier in full form. In such cases, not all of the information contained in these data sets would be available. Previous work in Named Entity Extraction has addressed this problem by first trying to find full PNPs, then looking for their abbreviated versions (Mikheev et al, 1998). However, we note that the current system can also recognize single-word unknown PNPs directly. Over 30% of the total PNPs used in these data sets are single-word PNPs, and in some categories, the number is much higher (e.g. 84% of place names are single words). Thus the ability of the classifier to handle both the presence and absence of common peripheral words in PNPs is being directly measured in our results.

[4] One thing we did manually correct was names that appear with their words in a non-standard order used for indexing, such as movie titles like *Ghost, The* and names like *Adams, John Quincy*. Each of these cases was restored to their "natural" word order.

4 Experimental Results

To assess the accuracy of our classifier, we ran three types of tests: *pairwise* tests of a single category against another single category, *1-rest* tests of a single category against the union of the other categories, and *n-way* tests, where all categories are against each other. The results of these tests are presented in Figure 1, sorted by classification accuracy and shown with standard deviations (computed from the ten separate train/test runs carried out for each result).

As expected, the n-way test was the most difficult. The ranking of results also reflects the inherent difficulty of the different categories. Overall, company names were most easily recognized, followed by drug names, person names, and place names, with movie titles proving the most difficult.

5 Discussion

In this section, we account for the experimental results above, analyze the contribution of each piece of our model to overall performance, and examine the various parameters learned during cross validation. We also present novel PNPs stochastically generated from our model, and conclude with demonstration of the ease with which this model can achieve proficiency with new categories.

5.1 Analysis of Experimental Results

Figure 2 shows the confusion matrix for the 5-way classification task. The area of each circle is proportional to the number of examples in that cell. Movies, places, and people are most often confused for one another, and drugs are often misclassified as places (they both contain many odd-looking one-word names). As an indication of the difficulty of dealing with movie titles, if the n-way tests are rerun as a four-way test without movie titles, average classification accuracy jumps from 88.1% to 93.2%. 3-way classification between companies, places, and people (similar to the ENAMEX classification task in MUC) is performed with an average classification accuracy of 94.6%.

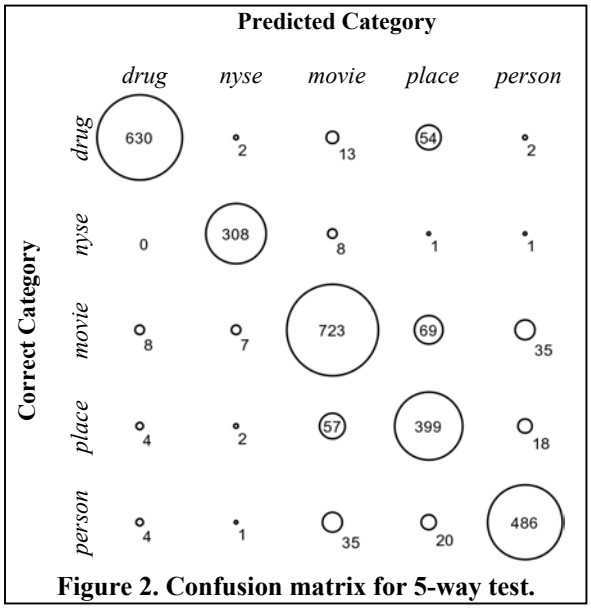

Figure 2. Confusion matrix for 5-way test.

The ease of identifying company names is largely attributable to the plethora of common words available, such as *International*, *Capital*, *Inc.*, *Corporation*, and so on. The difficulty of place names and movie titles is partly due to the fact that they contain words from many different languages (and thus the estimates learned blur together what should really be separate distributions). Movie titles are also the most inherently ambiguous, since they are often named after people (e.g. *John Henry*) or places (e.g. *Nuremberg*), and often contain words normally associated with another category (e.g. *Prozac Nation* and *Love, Inc.*). Mikheev et al (1998) report instances of similar ambiguity as a source of error in their work.

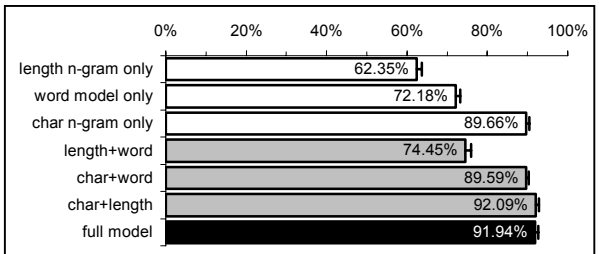

Figure 3. Classification accuracy for individual model components and combinations (4-way test).

A similar source of errors stems from words (and common intra-word letter sequences) that appear in one category and drive classification in other categories when there is insufficient information to the contrary. For example, in one run *Delaware* is erroneously classified as a company, because it was never seen as a place name, but it was seen in several company names (such as *GTE Delaware LP*). Cases like these appear to be an inherent limitation of this classifier. However we are being unusually restrictive by forcing our test set to be completely disjoint with our training set. In a real application, common place names like *Delaware* would have been trained on and would be readily recognizable as place names.

5.2 Contribution of Model Features

To assess the relative contribution of the three major components of our model (length n-gram, character n-gram, common words), we present accuracy results using each model component in isolation and each possible pair of components (Figure 3).[5] We use the 4-way *drug-nyse-place-person* test as a representative indicator of performance.

Each feature gives a classification accuracy significantly above a most-frequent-class baseline (34%), with the character n-gram being by far the most powerful single feature.

Combining features reveals that the character and length n-grams provide complementary information, but adding the word model to the character n-gram does not improve performance. The common word model by itself is quite effective, but it is largely subsumed by our high order character n-gram model, because common

[5] Note that the full model in Figure 3 is identical to the 4-way test in Figure 1. The slight difference in performance is merely due to data set differences.

short words are memorized as single n-gram entries, and common long words contain many common n-grams. The word model could be eliminated without hurting performance.

The word model could also be regarded as a reasonable baseline, since it is basically equivalent to the performance that could be expected from a (multinomial) Naïve Bayes word model, a model that is often used as a baseline in text classification tasks. As one further indicative baseline, we ran a publicly available variable n-gram language identifier on our data (Beeferman 1996). It achieves a performance of 76.54%. This is not a fair comparison: Beeferman explicitly notes that his system is unlikely to be reliable on very short inputs of the sort present in our data, but this nevertheless again shows that our system is sufficiently well tuned to the task at hand to achieve performance well above obvious baseline levels.

5.3 Impact of Other Model Parameters

In addition to the three major model components described above, performance is affected by the length of the n-gram models used, the use of a word length normalization constant for the character n-gram, and the amount of available training data.

5.3.1 Increasing N-Gram Length

The only important model parameters not set on held-out data are the sizes of the length and character n-gram models. In principle, the use of deleted interpolation with weights set on held-out data means that very large n-gram models could be used, and once data sparseness was a larger factor than predictive accuracy, the higher-order n-gram factors would be down-weighted. However in practice training and testing is exponentially slow in the length of the n-gram, and the largest useful n-gram size, once empirically determined, is relatively stable.

Table 1 shows classification accuracy of the character n-gram model alone for increasing values of n. Accuracy increases and plateaus, at which point increasing n further has no effect. The same analysis holds for increasing n for the word-length n-gram, also shown in Table 1.

n	1	2	3	4	5	6	7
char	60.0	81.4	87.7	89.2	89.5	89.7	89.8
length	46.4	60.1	62.2	62.4	62.4	-	-

Table 1. Classification accuracy of char and word-length n-gram models alone (4-way test).

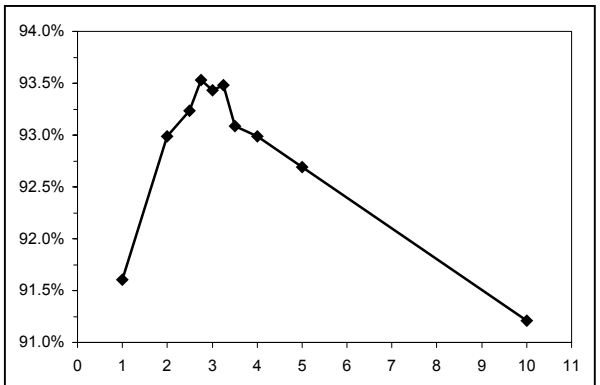

Figure 4. Classification accuracy vs. word length normalization constant (4-way test).

5.3.2 Word Length Normalization

As mentioned above, modeling words with a character n-gram model means treating characters as the unit of evidence, so that long words have more of an impact on classification than short words. But in many instances, it seems intuitively that words are a better unit of evidence (indeed, many telling common words like *Inc.* or *Lake* are very short). To compensate for this effect we introduce a parameter to normalize the probability assigned to each word in a PNP by taking the (*k*/*length*)'th root, where *length* is the number of characters in the word, and *k* is a global constant set with a line search on held-out data.

Figure 4 shows how varying the value of *k* affects performance on a typical run. The optimal value of *k* varies by data set, but is usually around 2 to 3. Probability judgments for words of length $<k$ are magnified, while those for words of length $>k$ are diminished. The result is that compelling short words can effectively compete with less compelling longer words, thus shifting the unit of evidence from the character to the word.

5.3.3 Training Data Size

Obtaining a large number of examples of each category to use as training data was not difficult. Nevertheless, it is still worth examining classifier performance as a function of the amount of training data provided.

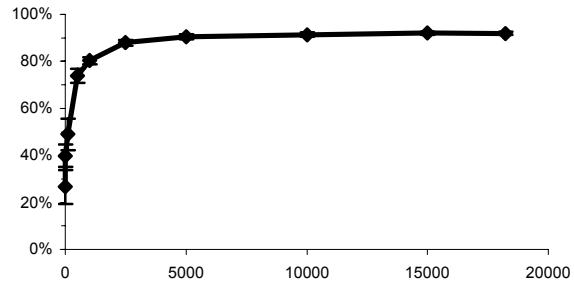

Figure 5. Classification accuracy vs. number of training examples (4-way test).

Figure 5 illustrates that while performance continues to improve as more training examples are provided, the classifier only requires a small subset of the training data to approach its full potential. This means that when faced with novel categories, for which large collections of examples are not immediately available, acquiring proficiency should still be possible.

Figure 5 also indicates that increasing the amount of training data would not significantly boost performance. This hypothesis is supported by the observation that the majority of misclassified examples are either inherently ambiguous, or contain words that appeared in another category, but that are not strongly indicative of any one category (as mentioned in Section 5.1).

5.4 Generation of Novel PNPs

Generative models are common for classification, but can also be used to perform generation. We stochastically generated a collection of novel PNPs as an alternative means for getting a sense of the quality of the learned models. A favorable selection of generated examples for each category is presented in Table 2.

drug*: Esidrine Plus Base with Moisturalent • Ambenylin • Carbosil DM 49*
nyse*: Downe Financial Grp PR • Intermedia Inc. • Host Manage U.S.B. Householding Ltd.*
movie*: Dragons: The Ever Harlane • Alien in Oz • El Tombre*
place*: Archfield • Lee-Newcastleridge • Qatad*
person*: Benedict W. Suthberg • Hugh Grob II • Elias Lindbert Atkinson*

Table 2. Sample of generated PNPs.

Not all generated examples are this coherent, but we are encouraged by the surprisingly natural look of a large fraction of what is generated. Sometimes entire training examples are generated, but usually the result is a mix of existing and novel words mixed together.

5.5 Generalization to New Categories

The fact that our classification technique is domain independent means that we can quickly attain high levels of performance on categories that have not previously been studied. We illustrate the versatility of our approach here, by applying the classifier to a novel domain, without making any changes to the classifier design.

Inspired by the game "Cheese or Disease?", featured on the MTV show *Idiot Savants* (see Holman, 1997), as our novel categories, we chose discriminating names of cheeses, and names of diseases. The basic idea is that there are many odd-sounding cheese and disease names, some of which are quite difficult to tell apart. This seemed like an ideal test for the PNP classifier.

Finding existing lists of cheeses and diseases on the web proved trivial (see Appendix for details), and since the classifier is domain independent, we were able to start training immediately. With 10 minutes of work, we had a classifier that achieved 93.5% classification accuracy. While recognizing references to cheese may not be high on the Defense Department's priority list, we feel that the ability to quickly reach high levels of proficiency in novel domains is a key benefit of our approach.

6 Conclusion

We have demonstrated that there are reliable regularities in the way names are constructed, which can be exploited for the purposes of named-entity classification. Specifically, there are common sequences of letters in the words used, there are common words that appear alongside uncommon words, and there are regularities in the number and length of words used in the name. These clues are sufficient for highly accurate classification, even in the absence of further context, and can be effectively used to complement existing context-based techniques for named-entity recognition by supplying an informed prior probability over categories.

It should not come as too much of a surprise that categories like drugs, companies, and movies have similarly constructed names. As Krauskfopf (2002) points out, coming up with new drug names is usually a multi-million dollar process in which special consultants are hired to find names that are different enough to be legally protected, but that have a "product of several powerful sounds" (*Prozac* is touted as one of the best invented drug names). It may be the case that our classifier is learning the regularities in drug names that these consultants have buried subconsciously in their heads. Generation of novel PNPs with this type of model could prove to be a compelling application on its own.

Acknowledgements

Special thanks to Stephen Patel, who worked on an earlier version of this project.

References

Abney, S. (1991). Parsing By Chunks. In Berwick, R., Abney, S., and Tenny, C., editors, *Principle-Based Parsing*. Kluwer Academic Publishers.

Appelt, D., Hobbs, J., Bear, J., Israel, D., Kameyama, M., Kehler, A., Martin, D., Myers, K., & Tyson, M. (1995). SRI International FASTUS system: MUC-6 test results and analysis. In *Proc. of the 6th Message Understanding Conference*, pp. 237-248.

Baluja, S., Mittal, V., & Sukthankar, R. (1999). Applying machine learning for high performance named-entity extraction. *Proceedings of the Conference of the Pacific Association for Computational Linguistics* (pp. 365-378).

Beeferman, D. (1996). Stochastic language identifier. Unpublished, Carnegie Mellon University. http://www.dougb.com/ident.html

Bikel, D., R. Schwartz, and R. Weischedel. (1999). An Algorithm that Learns What's in a Name. *Machine Learning* 34: 211–231.

Bikel, D.M., Miller, S., Schwartz, R. & Weischedel, R. (1997). Nymble: a High-Performance Learning Name-finder. *Proc. ANLP-97,* pp. 194-201.

Bodenreider, O., & Zweigenbaum, P. (2000). Identifying proper names in parallel medical terminologies. In *Medical Infobahn for Europe – Proceedings of MIE2000 and GMDS2000*, pp. 443-447.

Campbell, D. A., & Johnson, S. B. (1999). A Technique for Semantic Classification of Unknown Words Using UMLS Resources. In *AMIA '99 Annual Symposium* (American Medical Informatics Association), session on *Innovations in NLP*.

Cavnar, W. B. and Trenkle, J. M. (1994). Ngram Based Text Categorization. *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pp 161-169.

Charoenpornsawat, P., Kijsirikul, B., & Meknavin, S. (1998). Feature-Based Proper Name Identification in Thai. In *NCSEC-98 (The National Computer Science and Engineering Conference '98)*.

Collins M. and Singer Y. (1999). Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 189-196.

Dunning, T. (1994). Statistical identification of language. Computing Research Laboratory technical memo MCCS 94-273, New Mexico State University, Las Cruces, New Mexico.

Holman, C. (1997). TV on the edge: Idiot Box. *Creative Loafing, Atlanta, February 08, 1997*. <http://www.creativeloafing.com/archives/atlanta/newsstand/020897/b_edge.htm> (Visited: 3/20/02).

Krauskopf, L. (2002). Naming new drugs: Costly, complex. *The New Jersey Record, January 15, 2002*. <http://home.cwru.edu/activism/READ/Bergen011502.html> (Visited: 03/20/02).

Mikheev A., Moens M. and Grover C. (1999) Named Entity recognition without gazetteers. In *Proceedings of the Annual Meeting of the European Association for Computational Linguistics* (EACL'99), Bergen, Norway, pp. 1-8.

Mikheev, A. (1997). Automatic Rule Induction for Unknown Word Guessing. *Computational Linguistics* vol 23(3), ACL 1997. pp. 405-423.

Mikheev, A., Grover, C., & Moens, M. (1998) Description of the LTG System Used for MUC-7. *MUC-7*. Fairfax, Virginia.

Ramshaw, L. A. and Marcus, M. P. (1995). Text chunking using transformation-based learning. In Yarowsky, D. and Church, K., editors, *Proceedings of the Third Workshop on Very Large Corpora.*

Wacholder, N., Ravin, Y., & Choi, M. (1997). Disambiguation of Proper Names in Text. In: *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 202-208.

Appendix: Size and Source of Each Data Set

Category:	**drug** (6871 examples)
Description:	Micromedex 2000 USP Drug Index
Source:	my.webmd.com/drugs

Category:	**nyse** (3403 examples)
Description:	Companies on the NY Stock Exchange
Source:	www.nyse.com/listed
Category:	**movie** (8619 examples)
Description:	Internet Movie Database (IMDB) listing of 2000 movies and videos
Source:	us.imdb.com/Sections/Years/2000
Category:	**place** (4701 examples)
Description:	Collection of country, state/province, and city names from around the world
Source:	dir.yahoo.com/Regional/Countries
Category:	**person** (5282 examples)
Description:	List of people with online biographies
Source:	www.biography-center.com
Category:	**cheese** (599 examples)
Description:	Global database of cheese information
Source:	www.cheese.com
Category:	**disease** (1362 examples)
Description:	MeSH List of Diseases and Disorders
Source:	www.mic.ki.se/Diseases/alphalist.html