Periods, Capitalized Words, etc.

Andrei Mikheev * †
Harlequin Ltd. and HCRC

In this paper we present an approach which tackles three problems: sentence boundary disambiguation, disambiguation of capitalized words when they are used in positions where capitalization is expected and identification of abbreviations. All these tasks are important tasks of text normalization, which is a necessary phase in almost all text processing activities. The main feature of our approach is that it uses a minimum of pre-built resources. To compensate for the lack of pre-acquired knowledge, the system tries to dynamically infer disambiguation clues from the entire document itself. This makes our approach domain independent, closely targeted to each document and portable to other languages. We thoroughly evaluated our approach on the Brown Corpus and on a corpus of news wires articles from The New York Times. The system produced a very strong performance reaching about 99% accuracy on capitalized words and about 99.3-99.7% accuracy on sentence boundaries. This performance is the highest quoted in the literature for the tasks. We also present the results of applying our system to a corpus of news in Russian and training a part-of-speech tagger which uses a maximum entropy model that utilizes non-local features generated by our method.

1 Introduction

People who were involved in building real world text processing systems probably noticed that before one can start doing task-specific (interesting) development there is an annoying phase of text cleaning and normalization. What exactly is included into this phase depends on a particular architectural design and a breakdown of functionality, but as a minimum the documents should be segmented into so-called zones – paragraphs, tables, titles, etc. and then free-running text on these zones should be segmented into sentences. Sentence is considered as a central processing unit in the majority of NLP tasks: POS tagging, parsing, Information Extraction, Machine Translation, Text Alignment, Document Summarization, etc. The task of sentence boundary detection is closely linked to the task of capitalized words disambiguation the objective of which is to decide whether the first word of a sentence is a capitalized variant of a common word and hence should be treated in a case insensitive manner, or it is a proper name and hence should be processed with respect to its capitalization. In this paper we present a framework which treats sentence boundary detection and capitalized word normalization as parts of a single task which also includes detection of abbreviations in the text.

1.1 Sentence Boundary Disambiguation

Segmenting text into sentences in most cases is a simple matter – period, exclamation mark and a question mark usually signal a sentence boundary.

* Lismore House, 127 George Street, Edinburgh EH2 4JN, UK
† 2 Buccleuch Place, Edinburgh EH8 9LW, UK

© 1994 Association for Computational Linguistics
Example 1

*John Mackenzie lives in Dallas. This is a fact.*

However, there are cases when a period denotes a decimal point or is a part of abbreviation and thus it does not signal a sentence break. Furthermore, an abbreviation itself can be the last token in a sentence in which case its period acts at the same time as part of this abbreviation and as the end-of-sentence indicator (fullstop). Consider a narrative:

Example 2

*John Mackenzie Jr. lives in Dallas, Tex. This is a fact.*

Here we see three periods: the first one is part of the abbreviation “Jr.”, the second one is part of the abbreviation “Tex.” and at the same time sentence-end marker, and the last one (after the word “fact”) is a fullstop. A series of periods (…), ellipsis, can also occur within sentences or on sentence boundaries. Although question and exclamation marks are more stable, they still can be parts of names such as “Yahoo!” (name of a company) or “Which?” (name of a magazine).

As one can guess the most frequent source of ambiguity in the end-of-sentence marking is introduced by abbreviations: if we know that the word which precedes a period is not an abbreviation, then almost certainly this period denotes a sentence break. However, if this word is an abbreviation, then it is not that easy to make a clear decision. This is also amplified by the fact that abbreviations do not form a closed set i.e. one cannot list all possible abbreviations. It gets even worse – abbreviations can coincide with ordinary words i.e. “in” can denote the abbreviation for “inches”, “no” can denote the abbreviation for “number”, “bus” can denote the abbreviation for “business”, etc.

Even if we managed to robustly classify words into abbreviations and ordinary words, this would not solve the sentence boundary problem completely: an abbreviation can be sentence internal or it can be on a sentence boundary as exemplified in example 2. Obviously capitalization of the word which follows the period (or question/exclamation mark) is an important feature. This comes from the convention that in mixed-case texts a sentence should start with a capitalized word. Thus one can think of a rule that if a potential end-of-sentence marker is followed by a capitalized word there is a sentence break. This, however, is not as simple as that.

Capitalized words usually denote proper names – names of organizations, locations, people, artifacts, etc. – but there are special positions in the text where capitalization is expected. Such mandatory positions include the first word in a sentence, words in all-capitalized titles or table entries, a capitalized word after a colon or open quote, the first capitalized word in a list-entry, etc. Capitalized words in these and some other positions present a case of ambiguity – they can stand for proper names as in “White later said …”, or they can be just capitalized common words as in “White elephants are …”. Thus, when a capitalized word, which follows a potential end-of-sentence marker, is a common word that has been capitalized solely to signal the beginning of the next sentence – the disambiguation is clear. However, when such capitalized word denotes a proper name, its capitalization is not necessarily signals a new sentence. Consider these four examples:

Example 3

1. *Complaints should be sent to Dr. White.*

2. *At 3 p.m. Continental finalized its offer.*
3. *He stopped at Meadows Dr._ White Falcon was still open.*

4. *This happened at 3 p.m._ Continental finalized its offer.*

Here the periods in cases (1) and (2) are not end-of-sentence markers despite the next word is capitalized, and the periods in cases (3) and (4) are sentence breakers.

Figure 1 shows a simplified decision tree which classifies a period as sentence terminal or sentence internal, provided that the preceding and the following words have been disambiguated as abbreviations and proper names. There are three unambiguous\(^1\) combinations:

1. when a period follows a non-abbreviation and then is followed by a capitalized word – this period is a fullstop (example 1);
2. when an abbreviation is followed by a lowercased word (or a comma) - this period is abbreviation internal (example 2);
3. when an abbreviation is followed by a capitalized word which is disambiguated to be a capitalized common word (as opposed to proper name) – the period is sentence terminal (example 3.3).

There are two ambiguous outcomes. The first one is extremely infrequent – when a non-abbreviation is followed by a period and then by a lowercased word. This usually happens when the period or non-capitalization occurs accidentally. The second case represents a truly ambiguous outcome when an abbreviation is followed by a proper name. Sometimes such cases are sentence internal (examples 3.1 and 3.2) and sometimes they are sentence final (examples 3.3 and 3.4).

1.2 Disambiguation of Capitalized Words

As one can see the disambiguation of sentence breaking punctuation depends on whether or not the preceding word is an abbreviation and whether or not the following capitalized

---

\(^1\) Note that the abbreviations and capitalized words still present ambiguous tasks.
word is a proper name. Both of these tasks and especially the disambiguation of capitalized words are important tasks of text normalization. Let us look at the latter problem in more detail.

In general, the disambiguation of capitalized words (also called case normalization or proper name identification) in mixed-case texts does not seem to be too difficult: if a word is capitalized in an unambiguous position, e.g., not after a period or other punctuation which might require the following word to be capitalized (such as quotes or brackets), it is a proper name or part of a multi-word proper name. However, when a capitalized word is used in a position where it is expected to be capitalized (so-called mandatory position), for instance, after a period or in a title, our task is to decide whether it acts as a proper name or as a common word which was capitalized because of the mandatory convention. As we mentioned above a capitalized word which follows an abbreviation presents a case of double ambiguity——whether a period after the abbreviation is sentence terminal and thus the next word is in the mandatory position, and whether the capitalized word after the period is a proper name.

The disambiguation of capitalized words in the ambiguous positions leads to the identification of proper names and in this paper we will use these two terms and the term case normalization interchangeably. Note that this task, does not involve the classification of proper names into semantic categories (person, organization, location, etc.) which is the objective of the Named Entity Recognition task.

Disambiguation of capitalized words in mixed-case texts has hardly received much attention in the natural language processing and information retrieval communities, but in fact it plays an important role in many tasks. Many researchers observed that commonly used upper/lower case normalization does not necessarily help document retrieval. Church in (Church, 1995) among other simple text normalization techniques studied the effect of case normalization for different words and showed that “...sometimes case variants refer to the same thing (hurricane and Hurricane), sometimes they refer to different things (continental and Continental) and sometimes they don’t refer to much of anything (e.g. anytime and Anytime).” Obviously these differences are due to the fact that some capitalized words stand for proper names (such as Continental—the name of an airline) and some don’t.

Proper names are the main concern of the Named Entity Recognition subtask (Chinchor, 1998) of Information Extraction. There the disambiguation of the first word of a sentence (and in other ambiguous positions) is one of the central problems——about 20% of Named Entities are used in ambiguous positions. For instance, the word “Black” in the sentence-initial position can stand for a person’s surname but can also refer to the colour. Even in multi-word capitalized phrases the first word can belong to the rest of the phrase or can be just an external modifier. In the sentence “Daily, Mason and Partners lost their court case” it is clear that “Daily, Mason and Partners” is the name of a company. In the sentence “Unfortunately, Mason and Partners lost their court case” the name of the company does not involve the word “unfortunately”, but the word “Daily” is just as common a word as “unfortunately”.

Identification of proper names is also important in Machine Translation because normally proper names should be transliterated (i.e. phonetically translated) rather than properly (semantically) translated. In confidential texts, such as medical records, proper names must be identified and removed before making such texts available to unauthorized people. And in general, most of the tasks which involve text analysis will benefit from

2 This is not entirely true——some words derived from locations such as American, French, etc., are always written capitalized but in fact can stand for an adjective (American president) as well as a proper noun (he was an American).
the robust disambiguation of capitalized words in mandatory positions into proper names and capitalized common words.

1.3 This Paper
The disambiguation of capitalized words and sentence boundaries present chicken and egg problem: if we know that a capitalized word which follows a period is a common word, we can safely assign such period as sentence terminal. On the other hand, if we know that a period is not sentence terminal, then we can conclude that the following capitalized word is a proper name and also conclude that the preceding word-token is an abbreviation.

In this paper we present an approach which tackles sentence boundaries, capitalized words and abbreviations in a uniform way. The main feature of our approach is that it almost does not use any pre-built resources such as pre-trained statistical models, sets of disambiguation rules, detailed lists of words and phrases and so on. To compensate for the lack of pre-acquired knowledge, the system tries to dynamically infer disambiguation clues from the entire document. This makes our approach domain and genre independent and closely targeted to each document.

In the following sections we first discuss current state-of-the-art in the tasks in question, then we describe our uniform approach and present the results of evaluation of this approach on two corpora. We also present the results of porting our system to Russian and discuss the integration of our approach into a part-of-speech (POS) tagger.

In our experiments we used a corpus of 100 documents (64,337 words) from The New York Times 1996. This corpus was balanced to represent different domains and was used for the formal test-run at the 7th Message Understanding Conference (MUC’7) (Chinchor, 1998) in the Named Entity Recognition task. We used this corpus primarily because our text normalization system was originally built as part of a named entity (NE) recognition system (Mikheev et al., 1998(a)) for MUC’7 evaluation. The strong performance of our text normalization module proved to be one of the key factors for the whole NE system achieving a nearly human results at MUC’7. The other corpus we used in our work was the Brown Corpus (Francis & Kucera, 1982). This corpus of over one million words is composed from 15 sub-corpora which belong to different genres and domains, ranging from news-wire texts and scientific papers, to fiction and transcribed speech. The Brown Corpus is rich with out-of-vocabulary (unknown) words, spelling errors and ungrammatical sentences with complex internal format. Thus we could test how our method scales up and performs on different domains and genres.

2 Existing Approaches and Bottom Line Performance

Although, from the discussion in the previous section it is obvious that sentence boundary disambiguation is closely linked to the disambiguation of capitalized words and the identification of abbreviations, to our knowledge there were no previous approaches to tackle these problems simultaneously. For instance, part-of-speech taggers can deal with the disambiguation of ambiguously capitalized words but usually require already pre-tokenized text with identified sentence boundaries and abbreviations (Brill, 1995), (Ratnaparkhi, 1996) or include a sentence splitting component as a preprocessing tool which does not interact with the tagging process (Kupiec, 1992), (Mikheev, 1997(b)). Therefore, we are forced to present related work on each of the tasks in separate subsections after we, first.

3 In this paper we don't make distinction between words and word-tokens and use them interchangeably.
give a short summary of this review.

Throughout the review and in the paper in general we will provide performance measures for different systems and methods. A standard practice to measure the performance for the class of tasks we concerned with in this paper is to calculate two metrics:

\[
\text{accuracy} = \frac{\text{correctly assigned}}{\text{all assigned}} \quad \text{coverage} = \frac{\text{all assigned}}{\text{total candidates}}
\]

These two metrics are also often called precision and recall, which we think is a less appropriate way to call them in our case, because precision and recall were designed to compare assignments of sets of categories whereas in our tasks only one category is assigned. What we will measure in this paper is how accurately this single category is assigned and for how many such cases the assignment we can provide. We will also quote error rate which is the difference of the system accuracy and 100\% under the assumption that the coverage is 100\%.

Both sentence splitting and case normalization tasks are not extremely difficult. (Liberman&Church, 1992) claimed that it would not be too difficult to built an accurate sentence splitter for news-wire texts. A simple and naive algorithm applied to sentence splitting achieves under 7\% error rate on both MUC7 Corpus and the Brown Corpus.

A simple algorithm for case normalization achieves under 9\% error rate on both our corpora. The actual difficulty of these two tasks depends on the type of text: some, such as news-wire texts, contain a larger proportion of abbreviations and proper names which increases the number of ambiguous cases for sentence splitting and proper name disambiguation.

All existing approaches to sentence splitting use pre-acquired lists of abbreviations and the local context of a potential sentence boundary in terms of hand-crafted rules or statistical models induced from labelled examples. In general, rule-based systems are closely tailored to a particular corpus idiosyncrasies. The state-of-the-art systems achieve under 1\% error rate. There haven’t been much work done in developing and evaluating algorithms devoted solely for capitalized word disambiguation – this problem was always treated as part of a larger problem. Most commonly this problem is tackled by part-of-speech taggers which use local lexical and syntactic context encoded as statistical models or induced rewrite rules. The state-of-the-art systems achieve 3-4\% error rate on ambiguously capitalized words.

2.1 Existing Approaches to Sentence Boundary Disambiguation

2.1.1 Bottom-Line Performance Before embarking on the review of the existing approaches to sentence boundary disambiguation it is useful to assess the overall difficulty of the task. (Palmer&Hearst, 1997) suggested to estimate the lower-bound for the task by applying an extremely simple-minded algorithm which marks every potential sentence boundary punctuation as sentence terminal. They then adopted the UNIX STYLE program (Cherry&Vesterman, 1991) as the bottom-line performance. In our opinion the lower-bound algorithm is rather impractical for mixed-case documents and the UNIX STYLE algorithm is too knowledge intensive since it uses lists of abbreviations and a lexicon of known words.

Instead of estimating the lower bound and the bottom line as two different measures we suggest using so-called “period-space-capital letter” algorithm as a single bottom-line performance measure. This algorithm marks all periods/question and exclamation marks as sentence terminal if they are followed by at least one whitespace and a capital letter. It is extremely simple, does not use any lists or other resources and can be implemented as a short regular expression. We also extended the “period-space-capital letter” algorithm in a way that there can be optional brackets and quotes in between the period and the
capital letter so it will cover cased like “...soon (This...”. In terms of regular expressions this can be written as \[\[.!?\[\(\)"1+\[\[A/-Z/\].

The performance of sentence splitting algorithms not surprisingly depends on the proportion of abbreviations and proper names in the text and, hence, is domain and genre dependent – news-wire texts such as Wall Street Journal or New York Times will have a high proportion of both abbreviations and proper names whereas fiction will have a much lower proportion of them. We estimated the bottom-line performance on our two corpora and achieved 5.5% error rate on 2,787 potential sentence breaking punctuation of MUC’7 Corpus and 6.62% error rate on 49,532 potential sentence breaking punctuation of the Brown Corpus. Note, that internal periods of abbreviations (e.g. Y.M.C.A) or decimal points are correctly handled by the bottom-line algorithm and are not counted as potential sentence breakers. The performance of our bottom-line algorithm is comparable to that of STYLE which was reported to have 6.3% error rate (Palmer & Hearst, 1997) and about 2% higher than that of the lower-bound algorithm. It is also less sensitive than the lower-bound algorithm to the proportion of abbreviations in the text, because a considerable number of abbreviations is followed by a lowercased word or a comma (hence they are not sentence breaks) and this is handled well by the “period-space-capital letter” algorithm.

2.1.2 Rule Based Systems

The first large class of sentence boundary disambiguators uses manually built rules encoded usually in terms of regular expression grammars supplemented with lists of abbreviations, common words, proper names, etc. One such example is the UNIX STYLE program (Cherry & Vesterman, 1991) mentioned earlier. This program applies lists of abbreviations and proper names to the preceding and the following words and makes a decision based on a few predefined templates and heuristics. One such heuristic says that when a single capital letter is followed by a period, this period marks the end of sentence only if the next word is capitalized and is a function word. So the system will correctly not break a sentence in “...said C...” and correctly put a sentence break in “...vitamin C...”. As we already mentioned the error rate of this system was measured at about 6.3%.

The Alembic workbench (Aberdeen et al., 1995) contains a sentence splitting module which employs over 100 regular-expression rules written in Flex (Nicol, 1993). These rules make use of an extensive list of abbreviations which are classified into different groups according to their semantics and the following word expectation. For instance, the abbreviation “Mr.” is a honorific abbreviation which expects to be followed by a capitalized word, and “Ltd.” is a corporate designator which does not expect a capitalized word after itself. The error rate of this module was measured at 0.9% on a portion of Wall Street Journal used as the test-set in (Palmer & Hearst, 1997) when this system was equipped with lists pre-built specifically for this corpus.

(Palmer & Hearst, 1997) mention an unpublished work carried out at Mead Data Central, where a large finite state automata for sentence boundary disambiguation was developed over 9 person month. This system was tested on news-wire and legal mixed-case texts and achieved 99.7% accuracy. Unfortunately there is no publicly available information on details of this work and an interesting fact is that Mark Wason who was involved in the development of Mead’s sentence splitter participated in the development of EAGLE (Baldwin et al., 1997) text processing workbench where they employed a Maximum Entropy sentence splitter MAXTERMINATOR (Reynar & Ratnaparkhi, 1997) with performance around 98%. This suggests that Mead’s system is probably not portable enough to ensure it usability for unseen domains.

A rule-based sentence splitter which uses suffixes of words which surround potential sentence breaking punctuation is presented in (Muller et al., 1980). This system uses
suffix-based morphological analysis to filter out morphologically derived unknown words from possible abbreviation candidates. It was tried on English scientific abstracts and achieved 2.5% error rate but the measured abbreviation proportion in this texts was much higher than average.

2.1.3 Connectionists and Machine-Learning Systems  To develop a good rule-based system is a quite labor-consuming enterprise and since such systems are usually tailored to a particular corpus, they are not easily portable across domains. Automatically trainable software is generally seen as a way of producing systems quickly retrainable for a new corpus, domain or even for another language. There, however, is one catch - all known to us machine learning approaches to sentence boundary disambiguation require labelled examples for training, which implies a considerable investment into the annotation phase. Machine learning approaches represent sentence splitting as a classification problem with features such as words, suffixes, POS tags, etc. found in the local context of potential sentence breaking punctuation.

(Riley, 1989) reports on a decision tree classifier which was trained on 25 million words and achieved 0.2% error rate on the Brown Corpus. This system used context of one word to the left and one word to the right from a potential sentence splitting punctuation. This local context gave features such as length of a word, word capitalization, abbreviation class and probabilities for a word to appear at the beginning or at the end of a sentence. These probabilities were estimated from the words which start and end paragraphs rather than individual sentences, because in the training corpus paragraphs had unambiguous boundaries. This is why such a huge corpus (25 million words) was needed for training.

(Humphrey & Zhou, 1989) tried to use a feed-forward neural net to classify periods. The paper itself does not report on the accuracy of the approach but (Palmer & Hearst, 1997) claimed from personal communication with Joe Zhou that the neural network achieved unimpressive 7% error rate, similar to our bottom line performance.

First successful and easily reproducible results in applying a machine learning technique to the sentence splitting problem was reported in (Palmer & Hearst, 1997). Their system, SATZ, used part-of-speech distribution for words in the local context of a potential sentence splitting punctuation. These parts of speech were estimated from the unigram frequency of words vs. their syntactic classes with no respect to their capitalization. Then these estimates were fed into a back-propagation neural network and to a decision tree induction algorithm c4.5 (Quilan, 1986). The decision tree obtained 1.6% error rate on mixed-case texts and 1.9% on single case texts of a subset of Wall Street Journal. The neural net variant achieved 1.5% error rate on mixed-case texts and 3.3% error rate on single case texts. The SATZ system used a lexicon of 30,000 entries but produced a comparable result with the lexicon reduced down to 3,000 entries. The abbreviation list, however, proved to be more important; when the system operated without its abbreviation list of 206 entries, its error rate jumped to 4.9% on mixed-case texts. This system was then successfully ported to German and French with similar performance.

Maximum Entropy modelling was recently introduced into language engineering community and was successfully applied to the sentence splitting problems. We are aware of at least two such systems. (Reynar & Ratnaparkhi, 1997) report on MAXTERMINATOR, a system which used suffixes, prefixes and the abbreviation class of the words in the immediate local context of a period. It achieved 2.1% error rate on the Brown Corpus and 1.2% error rate on the subset of Wall Street Journal used by (Palmer & Hearst, 1997) in their evaluation. (Mikheev, 1998) reports on a similar system, LTFS, but with a sophisticated feature selection mechanism on top of maximum entropy parameter estimation. This system achieved 0.7% error rate on the same subset of Wall Street Journal.
Table 1

Distribution of proper names and common words among ambiguously capitalized word-tokens with respect to whether a word-token is a known or out-of-vocabulary word. Potential error cases for the lexicon lookup strategy are marked in bold.

|                  | **MUC’7 Corpus** |                  | **Brown Corpus** |
|------------------|-------------------|------------------|------------------|
|                  | all known unknown | all known unknown |                  |
| Total Words      | 2,695 2,047 648   | 48,193 42,326 5,867 |
| Proper Names     | 860 222 638      | 8,366 2,775 5,491 |
| Common Words     | 1,835 1,825 10   | 39,927 39,551 376 |

as (Palmer&Hearst, 1997) and (Reynar&Ratnaparkhi, 1997) and 1.3% on the entire Brown Corpus.

In general, the results obtained by (Riley, 1989) are the best ones quoted for the Brown Corpus so far, but the amount of required training data makes it impractical for the rapid development. It is also believed to be too closely targeted to the Brown Corpus which makes its use without retraining for other domains questionable. The SATZ system (Palmer&Hearst, 1997) has its strong point in that it is not very sensitive to capitalization and can work on single case texts. This system, however, requires estimated POS-tag probabilities for words which not always easily obtainable, and the performance of SATZ is still under 99%. The maximum entropy systems exhibited strong performance but they depend on capitalization and will require complete feature reengineering for being able to handle single case texts. Also as we mentioned above all these systems require pre-labeled examples for training and good lists of abbreviations.

2.2 Existing Approaches to Disambiguation of Capitalized Words

2.2.1 List Lookup Approach Research in Information Retrieval is well known for applying simple algorithms to text analysis – and in most cases their impact on the overall performance (Church, 1995) is at least comparable to that of knowledge intensive approaches. This can be partly explained by the fact that the operation unit for most of the IR tasks is the entire document rather than the individual occurrences of the words. In the disambiguation of proper names the majority of the systems in the IR field rely at best on pre-built gazetteers or lists of known names ((Hayes, 1994)).

Let us investigate one of the simplest strategies. The first obvious strategy for deciding whether a capitalized word in a mandatory position is a proper name or not is to apply lexicon lookup (possibly enhanced with a morphological word guesser, e.g., (Mikheev, 1997(a))). There we can mark as proper names all words which are not listed in the lexicon of common words. Let us investigate this strategy in more detail on MUC’7 Corpus of news-wire articles from The New York Times.

The 64,337-word corpus contained 2,695 ambiguously capitalized words, out of which 2,047 were listed in the lexicon of English common words. Ten common words were not listed in the lexicon and not guessed by our morphological guesser: “Forecasts”, “Benchmark”, “Everybody”, “Liftoff”, “Downloading”, “Pretax”, “Hailing”, “Birdbrain”, “Opting” and “Standalone”. In all our experiments we did not try to disambiguate between singular and plural proper names and we also did not count as an error the adjectival reading of words which are always written capitalized (e.g. American, Russian, Okinawan, etc.). The distribution of proper names among the ambiguously capitalized words is shown in Table 2.2.1.

Table 2.2.1 allows one to estimate the performance of the lexicon lookup strategy
which we take as the bottom-line. First, using this strategy we would wrongly assign the ten common words which were not listed in the lexicon. More damaging is the blind assignment of the common word category to the words listed in the lexicon: out of 2,047 known word-tokens 222 actually were used as proper names. This in total gives us 232 errors out of 2,695 tries – about a 8.6% misclassification error on ambiguously capitalized word-tokens.

We performed a similar experiment on the Brown Corpus. There a proportion of known to the lexicon common words in ambiguous positions was twice as high as in MUC'7 news-wire corpus. This made the task somewhat easier and reduced the overall error rate by about 1%: out of 48,193 tries we had 376 errors coming from unknown and unguessed words and 2,775 errors occurred when a known word was in fact used as a proper name. This in total gave us 7.4% error rate.

2.2.2 Part-of-Speech Tagging In the part-of-speech (POS) tagging field, the disambiguation of capitalized words is treated in the same way as the disambiguation of all other words. We are not aware of any specific studies on the performance of taggers solely on ambiguously capitalized words. This is probably because the impact of such words on the overall tagging performance is relatively small: the proportion of these word-tokens among all other word-tokens is only about 5% and with the 8% error rate of the bottom-line lexicon lookup strategy the overall contribution to the error rate is less than 0.4%.

Taggers of course can do much better than the lexicon lookup strategy because they take into account the immediate syntactic context for the words in question. However, as Church (1988) rightly pointed out “Proper nouns and capitalized words are particularly problematic; some capitalized words are proper nouns and some are not. Estimates from the Brown Corpus can be misleading. For example, the capitalized word “Acts” is found twice in Brown Corpus, both times as a proper noun (in a title). It would be misleading to infer from this evidence that the word “Acts” is always a proper noun.” Church then proposed to include only high frequency capitalized words in the lexicon and also label words as proper nouns if they are “adjacent to” other capitalized words. For the rest of capitalized common words he suggested that a small probability of proper noun interpretation should be assumed and then one should hope that the surrounding context will help to make the right assignment.

Local syntactic context, obviously, is more advanced than the list lookup approach. For instance, syntactic expectations strongly suggest proper noun reading in “Golden added...” and the adjectival reading in “Golden age of...”. However, not all ambiguously capitalized words are easily disambiguated by their surrounding part-of-speech context. For instance, many surnames are at the same time nouns or plural nouns in English and thus in both variants can be followed by a past tense verb. Capitalized words in the phrases “Sails rose...” or “Feeling himself...” can easily be interpreted either way and only knowledge of semantics disallows the plural noun interpretation of “Stars can read”.

Another challenge is to decide whether the first capitalized word belongs to the group of the following proper nouns or is an external modifier and therefore not a proper noun. For instance, “All American Bank” is a single phrase but in “All State Police” the word “All” is an external modifier and can be safely downcased. One might argue that a part-of-speech tagger can capture that in the first case the word “All” is followed by a singular proper noun (“Bank”) and hence is not grammatical as an external modifier and in the second case it is a grammatical external modifier since it modifies a plural proper noun (“Police”). However, one can list many similar cases such as “All American Games” where “All” modifies not the head noun “Games” but rather the adjective “American” and is part of a proper name.
The third challenge is of a more local nature – it reflects a capitalization convention adopted by the author. For instance, words which reflect the occupation of a person can be used in an honorific mode e.g. “Chairman Mao” vs. “ATT chairman Smith” or “Astronaut Mario Runko” vs. “astronaut Mario Runko”. When such a phrase opens a sentence, looking at the sentence only, even a human classifier has troubles in making a decision.

To evaluate the performance of part-of-speech taggers on ambiguously capitalized words we ran an HMM trigram tagger (Mikheev, 1997(b)) and the Brill tagger (Brill, 1995) on our two corpora. Since the taggers do not disambiguate abbreviations and sentence breaks themselves we simplified the task – the performance was measured only on ambiguously capitalized words which don’t follow abbreviations. For our task the mismatch between plural proper noun (NNPS) and singular proper noun (NNP) was not important, so we did not count this as an error. We also did not count as an error the adjectival reading of words which are always written capitalized (e.g. American, Russian, Okinawan, etc.). Both taggers used the Penn Treebank tag-set (Marcus et al., 1993).

To account for variations in tagger performance depending on the pre-trained model, we used two general models – one induced from the entire Brown Corpus and the other induced from the entire Wall Street Journal (WSJ). The performance of the two taggers was almost indistinguishable thus in Table 2.2.2 we present the results obtained only with the Brill tagger as more widely used.

We also tried to mix the models. A model for a part-of-speech tagger consists from two components – the lexicon of words with possible categories (POS tags) they can take on, and a contextual model which chooses one of these categories. The lexical component is also supplemented with preferences (probabilities) for a word to take one of its possible categories. To see the impact of each of these two components, we combined the two lexicons from the original models and used this combined lexicon with the contextual models of each of the two models (Brown Corpus and WSJ).

As can be seen in Table 2.2.2 in tagging ambiguously capitalized words the Brown Corpus presents a simpler task than a news-wire MUC’7 Corpus. This doesn’t come as a surprise – the proportion of proper names in news-wire texts is much higher. The first row of Table 2.2.2 shows the effect of overtraining – the model trained on the Brown Corpus performed very well on the same corpus (1.6% error rate) but performed very badly (6.7% error rate) on unseen texts of MUC’7 Corpus. The Wall Street Journal model performed on MUC’7 Corpus much better (3.8% error rate), probably because both MUC’7 Corpus and WSJ are of news-wire origin. However, the WSJ model performed not so great (4.7% error rate) on the Brown Corpus, assigning too many common words as proper names.

Table 2.2.2 shows that the lexicon plays the most important part in the model gen-

| Model                  | MUC’7 Corpus | Brown Corpus |
|------------------------|--------------|--------------|
|                        | error% -Prop| error% -Prop|
| Brown Corpus           | 6.7% 96     | 1.6% 254     |
| WSJ                    | 3.8% 40     | 4.7% 267     |
| Joint Lex + WSJ        | 3.4% 42     | 2.3% 216     |
| Joint Lex + Brown Corpus| 4.2% 57   | 2.2% 253     |
eralization. The same two models when used in conjunction with the lexicon which was produced by combining the lexicons from the individual models, performed much better on aggregate. The last row of Table 2.2.2 shows that when the Brown Corpus model was used with the combined lexicon, it produced much better results (2.5% cut in error rate) on unseen texts of MUC'7 Corpus while losing only some performance (0.7% increase in error rate) on Brown Corpus. The WSJ model benefited from the combined lexicon even more, scoring 3.4% error rate on MUC'7 corpus and 2.3% on the Brown Corpus.

This results basically confirm that the performance of a part-of-speech tagger is very dependent on how well it is tailored to a corpus and this in its turn depends largely on the lexicon in use. A very specific lexicon lists only a subset of possible POS tags for a word. Usually it lists only tags a word was seen with in the training data. For instance, the word “going” in general can act as a gerund or a noun but a corpus-specialized lexicon can list only the gerund category if no noun reading of “going” were seen in the training texts. This of course reduces the potential ambiguity for the tagger but when it is confronted with unseen texts it becomes too restrictive. A general lexicon which lists all or most possible POS tags for a word, can deal with unseen texts better, but it doesn’t produce such strong performance as the corpus-specialized lexicon on texts which come from a source similar to that seen in the training.

The taggers handled well the cases when a potential adjective was followed by a verb or adverb (“Golden added ..”) but they got confused when a potential noun was followed by a verb (VBP, VBZ, VBD), adverb (RB) or a preposition (IN): (“Butler/NNP was/VBZ ..” vs. “Safety/NN was/VBZ ..”), probably because the taggers could not distinguish between concrete and mass nouns. Not surprisingly the taggers did not do well on potential plural nouns and gerunds. The taggers also could not handle well the case when a potential adjective was followed by another capitalized word (“General/NNP Accounting/NNP Office/NNP” vs. “Gorgeous/JJ Doris/NNP Day/NNP”). In general, when the taggers did have strong lexical preferences, apart from the most obvious cases, they tended to assign a common word category to known capitalized words in the ambiguous positions. The performance of the part-of-speech tagging approach achieved about 3-4% error rate on aggregate which reduced the error rate of simple bottom-line strategy by half.

2.2.3 Lexico-Conceptual Patterns Unlike part-of-speech tagging, Information Extraction is mainly concerned with the identification and classification of proper names. Out of all named entities marked in MUC’7 Corpus, 20% were found in ambiguous positions. As can be seen in Table 2.2.1 the list-lookup method does not identify a quarter of the proper names. This translates into about 5% shortage in recall on the Named Entity recognition task. POS taggers do better – their error rate on the proper name subset of ambiguously capitalized words is reaching about 8%; this produces 1.6% shortage in recall. But POS taggers introduce other errors by wrongly tagging common words as proper nouns. This can instigate the development of spurious named entities which harms the precision on the Named Entity task. Furthermore, when a first word of a multi-word proper name is wrongly recognized (either attached or not included), it creates two errors at once – one in recall since the entire name is not identified and one in precision since a wrong (not entirely correct) name is marked. Not surprisingly, the task of Information Extraction and especially its subtask of Named Entity recognition requires more accurate methods to deal with identification of proper names.

In the Information Extraction field the disambiguation of ambiguously capitalized words was always tightly linked to the classification of the proper names into semantic classes such as person name, location, company name, etc. and to the resolution of coreference between the identified and classified proper names. This gave rise to the methods which aim at these tasks simultaneously. (Mani&MacMillan, 1996) describe a method of
using contextual clues such as appositives ("PERSON, the daughter of a prominent local physician") and felicity conditions for identifying names. The contextual clues (usually expressed in terms of lexico-conceptual patterns) themselves are then tapped for data concerning the referents of the names.

The advantage of this approach is that these contextual clues not only indicate whether a capitalized word is a proper name, but they also determine its semantic class. The disadvantage of this method is in the cost and difficulty of building a wide-coverage set of contextual clues and the dependence of these contextual clues on the domain and text genre. Contextual clues are very sensitive to specific lexical and syntactic constructions and the clues developed for the newswire texts are often not useful for legal or medical texts. Furthermore, the application of Lexico-Conceptual Patterns has limited coverage — only a subset of proper names is handled by them: Information Extraction aims at specific named entities and if a proper name does not belong to the subset of interest it is left unassigned.

2.2.4 Other Approaches: (Baldwin et al., 1997) mention a case normalization module as part of a text preprocessor included into EAGLE workbench. Incidentally this preprocessor includes MAXTERMINATOR sentence boundary detector (Reynar & Ratnaparkhi, 1997) mentioned above, but these two modules don’t talk to each other. The case normalization module uses a heuristic that a capitalized word in a mandatory position should be downcased if it is found lowercased in the same document. They also employ a database of bigrams and unigrams of lowercased and capitalized words found in unambiguous positions to make decisions on downcasing the sentence starting words and words in all-capital titles. This shows a departure from the traditional local context paradigm but unfortunately no performance evaluation and other details on this method are mentioned in the paper.

2.3 Shortened words: Abbreviations, Acronyms, etc.
Abbreviations and acronyms stand for shortened words and terms. There is a difference between acronyms and abbreviations. An acronym is usually formed by taking the first initials of a phrase or compounded-word and using those initials to form a word that stands for something. An abbreviation is formed by shortening a single word or by taking the first initials of a phrase or compounded-word but these initials do not form a legitimate English word. Thus LASER is an acronym for Light Amplification by Stimulated Emission of Radiation since “laser” is a legitimate word of English, but FBI is an abbreviation for the Federal Bureau of Investigation because there is no such English word as “fbi”.

Unfortunately, universally accepted standards for many abbreviations and acronyms do not exist. Customary usage within a certain professional group often determines when to abbreviate and which abbreviation to use: for instance, when engineers smash concrete samples into cubic centimetres, they use the abbreviation “cm3”, while doctors use the abbreviation “cc” when prescribing cubic centimetres of pain-killer. Although there were attempts to compile comprehensive dictionaries of abbreviations (Schwartz, 1955), this task does not look feasible with each professional field developing its own abbreviations and acronyms e.g. (UBerkeley, 1999).

There is no wealth of approaches to the induction of abbreviation reported in the literature. Usually these approaches rely on heuristics such as that single-word abbreviations are short and do not include too many vowels (e.g. Mr, Dr, etc.) and phrasal abbreviations and acronyms are written in all capital letters (e.g. FBI, NATO, LASER). It appears that there are no hard and fast rules for using periods in either acronyms or abbreviations. The practice seems more and more to drop the periods in the phrasal
abbreviations and acronyms whereas single word abbreviations are mostly written with the period.

All known to us sentence boundary detection programs rely on extensive lists of abbreviations. These abbreviations often are sorted into semantic classes such as honorific premodifiers (Mr, Prof), corporate designators (Ltd, Co), etc. More portable systems use an automatic induction of abbreviations from training corpora and do not rely on the semantic classification.

3 Our Approach, Required Resources and Intended Markup

3.1 Our Approach: General Description

As shown on figure 1 in order to solve the sentence boundary problem we have to solve two other problems. Therefore, our overall task can be decomposed (very crudely) into four parts:

- identify abbreviations and ordinary words in the text;
- disambiguate ambiguously capitalized words;
- assign unambiguous sentence boundaries (unfilled three branches of the decision tree on figure 1);
- in case when an abbreviation is followed by a proper name disambiguate sentence boundary;

The estimates from the Brown Corpus show that if we had in our disposal the entirely correct information on the first two tasks, then only 2,491 out of 49,532 potential sentence boundaries (i.e. 5%) would present the case of ambiguity. The same experiment on MUC7 Corpus gave us 95 out of 2,787 potential sentence boundaries i.e. 3.4%. The main problem is that when dealing with real world texts, we have to solve the abbreviation and proper noun tasks ourselves, which is far from trivial. Since the information from one of these subtasks can help to solve the other two, it makes sense to attempt to solve all these problems simultaneously.

As we discussed above, the bad news (well, not really news) is that virtually any common word can potentially act as a proper name or part of a multi-word proper name. The same applies to abbreviations: there is no closed list of abbreviations and almost any short word can be used as an abbreviation. Fortunately, there is good news too: important words are used in a document more than one time and in different contexts. Some of these contexts create ambiguous situations but some don’t. Furthermore, ambiguous things are usually unambiguously introduced at least once in the text unless they are part of common knowledge presupposed to be known by the readers.

This is an observation which can be applied to a broader class of tasks. For example, people are often referred to by their surnames (e.g. “Black”) but usually introduced at least once in the text either with their first name (“John Black”) or with their title/profession affiliation (“Mr. Black,” “President Bush”) and it is only when their names are common knowledge that they don’t need an introduction (e.g. “Castro,” “Gorbachev”). Thus our suggestion is to look at the unambiguous usage of the words in question in the entire document.

In the case of proper name identification we are not concerned with the semantic class of a name (e.g. whether it is a person name or location). We simply want to distinguish whether this word in this particular occurrence acts as a proper name (or part of a multi-word proper name) or it is just a common word which is capitalized because it is
used in a mandatory position. If we restrict our scope only to a single sentence, we might find that there is just not enough information to make a confident decision. For instance, *Riders* in the sentence “*Riders said later.*” is equally likely to be a proper noun, a plural proper noun or a plural common noun but if in the same text we find “*John Riders*” this sharply increases the proper noun interpretation and conversely if we find “*many riders*” this suggests the plural noun interpretation.

The above reasoning can be summarized as follows: if we detect that a word has been used capitalized in an unambiguous context (not in a mandatory position), this increases the chances for this word to act as a proper name in mandatory positions in the same document. And conversely if a word is seen only lowercased, this increases the chances to downcase it in mandatory positions of the same document. This, of course, is only a general principle and will be further elaborated elsewhere in the paper.

The same logic applies to abbreviations. Although a short word which is followed by a period is a potential abbreviation, the same word when occurred in the same document in a different context can be unambiguously classified as an ordinary word if it is used without trailing period, or it can be unambiguously classified as an abbreviation if it is used with a trailing period and is followed by a lowercased word or a comma.

### 3.2 Building Resources
One of the main objectives of our method is to be domain and genre independent and thus reusable without retraining or fine-tuning for new texts of unknown origin. An important consideration is that the method should be easily implementable and efficient. Thus instead of building detailed resources such as statistical models, sets of disambiguation rules, detailed lists of words and phrases and so on, our approach relies on an absolute minimum of preexisting resources. To compensate for the lack of pre-acquired knowledge, the system tries to dynamically infer disambiguation clues from the entire document as was outlined above.

There are only four word lists which are used by our method. The first one is the list of common words. There is a variety of such lists for many languages already available (Burnage, 1990), and usually words in such lists are also supplemented with morphological and part-of-speech information. However, we don’t need to rely on preexisting common word lists - such a list can be easily automatically obtained by collecting and counting words from raw texts. To smooth for potential spelling and capitalization errors we set a threshold on a frequency of occurrence for words and put in our list only the words which occurred at least three times in the texts. For the development of the resources we used texts of about 300,000 words from The New York Times, supplied as training data for MUC’7. These texts of course did not include our MUC’7 Corpus.

The second list that we use is a list of common words which most frequently are seen in mandatory positions. This list can be easily obtained too. Remember, that the lexicon lookup strategy for capitalized word disambiguation (section 2.2.1) performs with less than 9% error rate. We used our general word list described above, over our training texts to tag capitalized word in mandatory position as common words and as proper names, then we collected and counted these words and put in the list 200 most frequent sentence starting common words. As one can guess the most frequent sentence starting word was “the” and the second most frequent one was “he”. The list included a few of adverbs such as “however”, “suddenly”, “once” etc., some prepositions such as “in”, “to”, “by” and even a few verbs – “let”, “have”, “do”, etc.

The third list is a list of proper names which we expect to see often in mandatory positions. The list lookup approach gave us only a part of such words because it never tags a known word as a proper name. To compensate for this we added to the frequent proper name list the words which are frequently used in our training texts as single capitalized
words in unambiguous positions. For instance the word “The” can be frequently seen
capitalized in unambiguous positions but it is always followed by another capitalized
word—thus we don’t count it as a candidate. Doing this we produced a list of 200 most
frequent proper names.

From the resources found on the Internet we compiled a relatively short list of 270 ab-
breviations which included honorific abbreviations (Mr, Dr), corporate designators (Ltd,
Co), month name abbreviations (Jan, Feb), abbreviations of American states (Ala, Cal)
and some measure unit abbreviations (ft, kg). Although we described these abbreviations
in groups, we did not encode this information in the list—the only information it provides
is that a word is an abbreviation. As we will show later this list is not absolutely crucial
to our approach and the system can successfully operate even without it.

An important thing is that it is very easy to specialize the above describe lists for a
particular domain. Using the existing resources our system tags proper names, common
words and abbreviations with a high accuracy. Note that the system uses these lists only
as a supplementary tool and tries to dynamically infer information about the unknown
words from the document itself. The results of our system performance can be used
to recompile the lists of most frequent sentence starting common words, most frequent
proper names and the list of abbreviations for a new domain.

3.3 Corpora Markup for Evaluation

We already presented MUC’7 Corpus and the Brown Corpus (section 1.3) as the two
corpora we use in our experiments. MUC’7 Corpus consists of a 100 documents, each
of which is made of a dozen of paragraphs of five-six sentences. We tokenized the text
of the paragraphs into tokens represented as XML elements using an XML-aware FSA
tokenizer of English (Mikheev et al., 1998(b)). We treated periods and all other potential
sentence breaking punctuation as first-class citizens and adopted a convention to always
tokenize a period (and other punctuation) as a separate token when it is followed by a
whitespace or a line-break. We then applied our system to MUC’7 Corpus and manually
corrected its output to produce the “golden standard” text. We marked all capitalized
words in mandatory positions as proper names (<W T="NNP">Moscow</W>) or common
words (<W T="com">The</W>), we marked all words with trailing periods as abbreviations
(<W T="NN">e.g</W>) or ordinary words (<W T="N">soon</W>) and we marked potential
sentence breaking punctuation as fullstops (T=’.’), part-of-abbreviation (T=’A’) or both
(T=’*’). An example of such markup is displayed on Figure 2.

In our experiments we also used the POS tagged version of the Brown Corpus dis-
tributed as part of Penn Treebank (Marcus et al., 1993). We had to convert the text
from the original format to our XML format. The Brown Corpus contains information
about paragraphs and sentences, it provides a POS tag for each word so it can easily be
mapped into whether a word is a proper name or not. The only change we had to make
was the treatment of periods and abbreviations. In the Brown Corpus abbreviations were
tokenized together with their trailing periods whereas in our markup periods should be
tokenized as separate tokens. Periods in the Brown Corpus are unambiguous whereas in
our markup they can take on one of the three tags (T=’.’, T=’A’ or T=’*’). Therefore,
we had to split the final periods from the abbreviations and assign them with T=’A’ and
T=’*’ tags according to whether the abbreviation was in the middle of a sentence or it
was the last token in a sentence. There were also quite a few infelicities in the original
tokenization and tagging of the Brown Corpus which we corrected by hand.

When converting the Brown Corpus to our XML format, naturally we hid the informa-
tion about sentences from our system but it was fully recoverable for the evaluation
purposes. We also noticed that sometimes when an abbreviation was used at the end
of a sentence, this sentence was closed with an additional period e.g. “Tex. This” and
sometimes it was not the case. While in the first case there is virtually no ambiguity,
the second case is ambiguous. Thus we decided to strip additional fullstops after abbre-
viations to produce more truly ambiguous cases. For the same reason we also decided
to strip the paragraph information – to provide more challenges for our system. If an
abbreviation is used at the end of a paragraph – there is no ambiguity – its period acts
as a sentence-end marker. So we converted such cases that a paragraph-ending sentence
is directly followed by the first sentence of the next paragraph.

In all our experiments we treated embedded sentence boundaries in the same way
as normal sentence boundaries. The embedded sentence boundary occurs when there
is a sentence inside a sentence. This can be a quoted direct speech sub-sentence inside
a sentence, this can be a sub-sentence embedded in brackets, etc. We consider closing
punctuation of such sentences equal to closing punctuation of ordinary sentences and
indeed this makes easier to find the starting tokens of the embedded sentences.

4 Getting Abbreviations and Potential Abbreviations

The answer to the question whether or not a word token is an abbreviation solves largely
the sentence boundary problem. In the Brown Corpus 92% of potential sentence bound-
daries come after a regular word. MUC’7 Corpus is richer with abbreviations and this
totals to 83%. As we discussed earlier there exists a variety of abbreviation lists but these
lists are very likely not to cover all abbreviations in a text especially in a specialized tech-
nical domain. This means that we cannot simply apply a heuristic which says that if a
period follows a word which is not listed in our abbreviation list, this period is a fullstop.

Since we cannot assume that we will be able to identify all the abbreviations in the
text, we have to develop a more robust technique. Basically at this initial stage we are
interested in identifying non-abbreviations, since this allows us to assign a substantial
proportion of sentence boundaries. Abbreviations and non-abbreviations form two mutually
exclusive sets, but we can expect a certain proportion of word-tokens to fail of being
robustly classified to either of the two categories i.e. to fall in between the two classes.
We call such word-tokens potential abbreviations.

Let us first start with abbreviations. We already mentioned in section 3.2 that we
collected a list of 270 abbreviations. When we applied the abbreviation list lookup to
MUC7 Corpus it achieved a high accuracy (1.6% error rate) but covered only 64% of
abbreviations. On the Brown Corpus the list lookup achieved a nearly 100% accuracy
but again it managed to cover only 62% of abbreviations. A natural extension here is to
supplement the list-lookup with guessing heuristics which use the surface lexical form of a
word-token:

- a word-token is an abbreviation if it consists of only consonants, followed by a
Table 3  
Abbreviations and ordinary words assigned by combinations of three strategies.

| Method                        | MUC’7 Corpus | Brown Corpus |
|-------------------------------|--------------|--------------|
|                              | Abbr. Words  | Abbr. Words  |
| Abbreviation list +           | accuracy     | 98.5% 99.95%| 99.8% 100% |
| Surface guessing              | coverage     | 98.3% 97.7% | 99.7% 99.1% |
| Abbreviation list +           | accuracy     | 98.5% 99.95%| 99.9% 100% |
| Surface guessing +            | coverage     | 98.3% 97.8% | 99.9% 99.7% |
| Positional guessing           | accuracy     | 99.5% 99.8% | 99.9% 99.9% |
| Surface guessing +            | coverage     | 91.2% 97.7% | 92.5% 99.6% |

The surface guessing heuristics on their own achieve much better performance than that of the abbreviation list: their accuracy was nearly 100% and their coverage was 88% on MUC’7 Corpus and 90% on the Brown Corpus. When the surface guessing heuristics were applied together with the abbreviation list, we achieved a very strong performance as displayed in the “Abbr.” columns of the first row of table 4. Only one abbreviation from MUC’7 Corpus and only three abbreviations from the Brown Corpus failed to be identified. The system also incorrectly assigned as abbreviations seven ordinary words in MUC’7 Corpus and eight ordinary words in the Brown Corpus. This totals to 98.4% in accuracy and coverage on MUC’7 Corpus and to 99.7% on the Brown Corpus.

The surface guessing heuristics supplemented with a list of abbreviation are accurate but they still can miss some abbreviations. For instance, if an abbreviation like “sec.” or “Okla.” are not listed in the list of abbreviations, the guessing rules will not uncover them. Basically any short word followed by a period can act as an abbreviation but most likely such a word will not be in the list of common words. Therefore, we introduced a notion of potential abbreviation: this is a short word when it is followed by a period and not found in the list of common English words. It is not safe to make a sentence boundary decision on a period which follows a potential abbreviation without having some additional information. Thus, the number of potential abbreviations together with found abbreviations determine the number of cases in which we can not safely assign a sentence boundary as a fullstop.

We can safely assign as fullstops the periods which follow only those word-tokens which has been identified neither as abbreviations nor as potential abbreviations. Column “Words” of the first row of table 4 shows that the abbreviation lists together with the surface guessing heuristics (and supplemented with the potential abbreviations identifier) manage to discriminate ordinary words with almost perfect accuracy (99.93% and 100% on our two corpora). However about 1-2% of the ordinary words could not been identified safely enough and thus were classified in the intermediate set of potential abbreviations.

4 All-capitalized word-tokens can stand for acronyms or phrasal abbreviations which are usually written with no period.
Naturally, we would like to reduce the proportion of potential abbreviations without sacrificing the accuracy. In other words we can try to disambiguate some of the potential abbreviations and classify them as definite abbreviations or definite non-abbreviations. Here we apply a simple technique which looks in the entire document for contexts a potential abbreviation is used in. If a potential abbreviation is used elsewhere in the document without a trailing period, we can conclude that this in fact is an ordinary word. To decide whether a potential abbreviation is in fact an abbreviation we look for the contexts when it is followed by a period and then by a lowercased word, a number or a comma. We call this the *positional guessing strategy* (PGS).

For instance, the word “Kong” followed by a period and then by a capitalized word cannot be safely classified as an ordinary word and therefore it is a potential abbreviation. But if in the same document we detect a context “lived in Hong Kong in 1993” this indicates that “Kong” is normally written without a trailing period and hence is not an abbreviation. Having established that, we can apply this findings to the non-evident contexts and classify “Kong” as an ordinary word throughout the document. However, if we detect a context as “Kong, said” this indicates that “Kong” is normally written with a trailing period and hence is an abbreviation. This gives us ground to classify “Kong” as an abbreviation in all its occurrences within the same document.

The positional guessing strategy relies on the assumption that there is a consistency of writing within the same document. Different authors can write “Mr” or “Dr” with or without trailing period but we assume that the same author (the author of a document) will write it in the same way consistently. This also applies to the cases when an abbreviation coincides with an ordinary word. In general, it is unlikely that both “Sun, (as for Sunday) and “Sun” (the name of a newspaper) will appear in the same document.

There indeed can occur a situation when the same potential abbreviation is used as an ordinary word and as an abbreviation within the same document. To tackle this, we collect bigrams of potential abbreviation with their preceding words when they are used in unambiguous contexts as explained earlier. Now we can assign ambiguous instances on the basis of these bigrams. For instance, if in our document we found a context “vitamin C is” we assign “C” in this context as non-abbreviation, then we store the bigram “vitamin C” and when we see such bigram in an ambiguous position “vitamin C. Research” we can disambiguate “C” as non-abbreviation. If in the same document we also detected a context “John C. later said” we store the bigram “John C.” and can assign “C” as abbreviation in an ambiguous context like “John C. Research”.

The second row of table 4 shows the results when we supplemented the abbreviation list and the surface guessing heuristics with the positional disambiguation strategy. Its impact on MUC77 corpus was not that great – it managed to disambiguate only 8 potential abbreviations as ordinary words. On the Brown Corpus its contribution was much higher: it disambiguated 271 potential abbreviations as ordinary words which boosted the coverage on ordinary words from 99.1% to 99.7%. This strategy also corrected 4 wrongly assigned abbreviations and disambiguated 1 more potential abbreviation as an actual abbreviation.

We also measured the performance of our abbreviation detection approach when no abbreviation list was available. The third row of table 4 shows that the surface guessing heuristics with the positional disambiguation strategy and without the abbreviation list achieve quite a decent performance. First the positional strategy boosted the coverage on abbreviations (column “Abbr.”) by about 2% in comparison with the guessing heuristics only. In comparison to the full fledged system the coverage drops by about 7% but still stays very competitive (91.2% and 92.5% on our two corpora). We also see a slightly improved accuracy on abbreviations. This is because some of the abbreviations listed in the abbreviation list coincided with ordinary words and sometimes forced a wrong
classification.

As for ordinary words – the results of the no-list system are virtually indistinguishable from the full fledged system. This allows us in both cases to assign sentence boundaries with 100% accuracy to about 92% of potential sentence boundaries on the Brown Corpus and to 83% on MUC’7 Corpus. Note also that not all periods after abbreviations are ambiguous. Abbreviations which are followed by a lowercased word, number or a comma are most certainly not at the end of a sentence, hence their periods are not sentence terminal. In the Brown Corpus there are only a few such cases (2%) but in MUC’7 Corpus massive 12% of periods fall in this category. This makes the proportion of truly ambiguous potential sentence breaking punctuation to be around 5-6% in both our corpora.

5 A Knowledge Free Method for Proper Name Disambiguation

The classification of text-tokens into ordinary words and potential abbreviations and handling of non-ambiguous abbreviations allowed us to assign about 94-95% of potential sentence boundaries with almost 100% accuracy. The second key task of our approach is the disambiguation of capitalized words which follow a potential sentence boundary punctuation. Apart from being an important task of text normalization, the information about whether or not a capitalized word which follows a period is a common word allows us to accurately assign as sentence terminal another 1-1.5% of the remaining 5-6% of unassigned potential sentence boundaries.

Our study in section 2.2.2 revealed that POS taggers achieve about 3-4% error rate on this task but they do not disambiguate all the cases. Since all the taggers we know presuppose that the text is split into sentences and that abbreviations are unambiguously tokenized, a capitalized word after an abbreviation inside a sentence does not present a case of ambiguity – it is a proper noun. Unlike POS taggers we cannot presuppose that the text is split into sentences, thus, every period acts as a potential sentence breaker and creates a potential mandatory position for capitalized words.

We tackle capitalized words in a similar fashion as we tackled the abbreviations – by the analysis of the distribution of ambiguous words in the entire document. This is implemented as a cascade of simple strategies which was briefly described in (Mikheev, 1999).

5.1 The Sequence Strategy

Our first strategy for the disambiguation of ambiguously capitalized words is to explore sequences of words built from the contexts where the same words are used unambiguously with respect to their capitalization. We call it the Sequence Strategy. The rationale behind this is that if we detect a phrase of two or more capitalized words and this phrase occurs in an unambiguous position we can be reasonably confident that even when the same phrase starts from an unreliable position all its words still have to be grouped together and hence are proper nouns. Moreover, this applies not just to the exact replication of such a phrase but to any partial ordering of its words of size two or more preserving their sequence.

For instance, if we detect a phrase “Rocket Systems Development Co.” starting from an unambiguous position (in the middle of a sentence), we memorize it and also we memorize its sub-phrases “Rocket Systems,” “Rocket Systems Co.,” “Rocket Co.,” “Systems Development,” etc. If then in the same document we see “Rocket Systems” in a mandatory position (after a period) we assign the word “Rocket” as a proper noun because it is

5 some taggers require a tokenized input and others produce tokenization at a preprocessing step.
a part of a multi-word proper name which we saw in unambiguous context. A span of capitalized words can also internally include lower-cased words of length three or shorter, alphanumerals and symbols. This allows us to capture phrases like “A & M”, “The Phantom of the Opera”, etc. We generate partial orders from such phrases in a similar way but insist that every generated sub-phrase should start and end with a capitalized word.

We can also apply the Sequence Strategy to common words by collecting bigrams of the lower-cased words of the document with their following words. For instance, from a context “continental suppliers of Mercury” we generate three bigrams: “continental suppliers”, “suppliers of” and “of Mercury”. We don’t attempt to build longer sequences and their partial orders because we cannot in general restrict the scope of dependencies in such sequences. Now, when we see a context “Continental suppliers” in a mandatory position, we can use the fact that we collected the bigram “continental suppliers” where the word-token “continental” was unambiguously a common word. On this basis we can assign ambiguously capitalized word-token “Continental” as a common word.

The Sequence Strategy is extremely useful when dealing with names of organizations since many of them are multi-word phrases composed from common words. And indeed, as is shown in Table 5.5, the accuracy of this strategy when applied to proper names was about 99% with coverage of about 8.9%. Out of 860 ambiguously capitalized proper names of MUC’7 Corpus (table 2.2.1), 75 were marked and only one of them incorrectly. On the Brown Corpus this strategy marked wrongly only 6 of 710 cases. All these wrong assignments came as a result of erroneous capitalizations in the documents but the high accuracy confirms that this strategy is not oversensitive to such errors.

For tagging common words the Sequence Strategy was 100% accurate covering 16.7% of ambiguously capitalized common words on MUC’7 Corpus and 25.5% on the Brown Corpus. In total on both proper names and common words the Sequence Strategy achieved 99.7-99.8% accuracy and the coverage of 14.2% on MUC’7 Corpus and 22.5% on the Brown Corpus.

5.2 Frequent List Lookup Strategy
Our next strategy is to mark words from the frequent starter lists (section 3.2). Ambiguously capitalized words found in the list of frequent starting common words are marked as common words and words found in the list of frequent proper names are marked as proper names. Note, however, that the Sequence Strategy is applied prior to the frequent list assignment and thus a word from one of these lists is not always will be marked according to its list class. Among such cases resolved by the Sequence Strategy were a multi-word proper name “To B. Super” where both “To” and “Super” were correctly identified as proper nouns and a multi-word proper name “The Update” where “The” was correctly identified as part of the magazine name. Both “To” and “The” were listed in the frequent list and therefore were very implausible to classify as proper nouns but nevertheless the system handled them correctly.

The Frequent List Lookup Strategy is extremely accurate (99.8-100%). The only few wrong assignments were in cases like “Mr. A”, “Mrs. Someone” and words in titles like “I’ve got a Dog” where “A”, “Someone” and “I” were assigned as common words but they were tagged as proper nouns in the Brown Corpus.

The Frequent List Lookup Strategy is not very effective for proper names where it covers only about 6-7% but it is extremely effective for common words where it covers about 60-70% of ambiguously capitalized common words. In total (on both categories) this strategy achieves almost perfect accuracy with the coverage of about 50% as can be seen in Table 5.5.
5.3 Single Word Assignment
Like the Sequence Strategy, our next strategy also uses information from the entire document. We call this strategy Single Word Assignment, and it can be summarised as follows: if we detect a word which in the current document is seen capitalized in an unambiguous position and at the same time it is not used lowercased, this word in this particular document, even when used capitalized in ambiguous positions, is very likely to stand for a proper name as well. And conversely, if we detect a word which in the current document is used only lowercased, it is extremely unlikely that this word will act as a proper name in an ambiguous position and thus, such a word can be marked as a common word.

Note, that by that time the Sequence Strategy and the Frequent List Strategy have been already applied, and all high frequency sentence-initial words were recognized as common words. This ordering is important because even if such a word is observed in a document only as a proper name (usually as part of a multi-word proper name), it is still not safe to mark it as a proper name in ambiguous positions. Note, however, that these words can be still marked as proper names (or rather as parts of proper multi-word names) by the Sequence Strategy.

Table 5.5 shows the success of the Single Word Assignment strategy. It marked 470 proper names in MUC7 Corpus from which only 1 was marked incorrectly, and it marked 66 common words from which 2 were marked incorrectly. The only word which was incorrectly marked as a proper name was the word “Insurance” in “Insurance company...” because in the same document there was a proper phrase “China-Pacific Insurance Co.” and no lowercased occurrences of the word “insurance” were found. The two words incorrectly marked as common words were: “Trade” in “Trade Representation office...” and “Satellite” in “Satellite Business News” – in both cases these words were seen only lowercased in the corresponding documents. Five out of ten words which were not listed in the lexicon (“Pretax”, “Benchmark”, “Liftoff”, “Downloading” and “Standalone”) were correctly marked as common words because they were found to exist lowercased in the text.

One notable difficulty for the Single Word Assignment represent words which denote profession/title affiliations. These words modifying a person name might require capitalization – “Sheriff John Smith”, but in the same document they can appear lowercased – “the sheriff”. When the capitalized variant occurs only as sentence initial, our method predicts that it should be decapitalized. This, however, is an extremely difficult case even for human indexers – some writers tend to use certain professions such as Sheriff, Governor, Astronaut, etc., as honorific affiliations and others tend to do otherwise. A generally difficult case for Single Word Assignment is when a word is used as a proper name and as a common word in the same document, and especially when one of these usages occurs only in an ambiguous position. For instance, in a document about steel the only occurrence of “Steel Company” happened to start a sentence. This lead to an erroneous assignment of the word “Steel” as common noun. Another example: in a document about “the Acting Judge”, the word “acting” in a sentence “Acting on behalf...” was wrongly classified as a proper name. These difficulties, however, often are compensated by the Sequence Strategy which is applied prior to the Single Word Assignment and can tackle such cases.

The Single Word Assignment was very useful for proper name identification: it achieved accuracy of over 99% and covered 50.8% of ambiguously capitalized proper names in MUC7 Corpus and 35.9% in the Brown Corpus. On common words this method was less accurate (about 97%) covering 3-4% of the cases.
5.4 Quotes, Brackets and “After Abbr.” Heuristic
Capitalized words in quotes and brackets do not directly contribute to our major task of sentence boundary disambiguation but they still present a case of ambiguity for the task of capitalized word disambiguation. To tackle this we implemented a few simple heuristics:

- if a capitalized word is in short quotes or brackets it is a proper noun: e.g. John (Cool) Lee;
- if there is a lowercased word, a number or a comma which is followed by an opening bracket and then by a capitalized word – this capitalized word is a proper noun: e.g. “...happened (Moscow News reported yesterday) but...”;

These heuristics are reasonably accurate – they achieved over 98% accuracy on both our corpora but they covered only about 6-7% of proper names.

When we studied the distribution of capitalized words after capitalized abbreviations, we uncovered an interesting empirical fact. A capitalized word which follows a capitalized abbreviation is almost certainly a proper name unless it is listed in the list of frequent sentence starting common words (see section 3.2) i.e. it is not “The”, “However”, etc. The accuracy of this heuristic is 99.2% and not surprisingly in 99.5% of cases the abbreviation and the following proper name belonged to the same sentence. We will use this fact later on when we deal with sentence boundaries. Naturally, the coverage of this “After Abbr.” heuristic depends on the proportion of capitalized abbreviations in the text. In the Brown Corpus this heuristic disambiguated 10.5% of ambiguously capitalized proper names and in MUC’7 Corpus it covered 29.7% of such cases.

5.5 The Overall Performance
Table 5.5 displays the results of the application of the described strategies to our two corpora. In general our method achieved 99.85% accuracy with 87.6% coverage on MUC’7 Corpus and 99.7% accuracy with 91.7% coverage on the Brown Corpus. This means that our method assigns nearly perfectly 9 out of 10 ambiguously capitalized word-tokens and leaves one capitalized word-token unassigned. When we concentrate on the impact of the individual strategies, we see that for the proper name category the most important is the Single Word Assignment then “After Abbr.” strategy and then the Sequence Strategy. For common words the most important is the Frequent List strategy and the Sequence Strategy.

Now we have to decide what to do with the remaining 10% of unassigned ambiguously capitalized word. If we apply the lexicon lookup strategy (section 2.2.1) to the unassigned words we will get about 8% error rate which will translate into a tradeoff of dropping 0.8% in accuracy for gaining 100% coverage. This makes the overall accuracy to be around 99%. If we apply a POS tagger to these words – this reduces the error rate on unassigned words by half and the system performance reaches 99.4% in accuracy with 100% recall.

Another direction to reduce the number of unclassified cases before applying a POS tagger or the lexicon lookup strategy is to share information between documents. To share individual words is a bit risky but word sequences are quite stable across documents. Thus we can accumulate multi-word proper names and lowercased word bigrams across several documents. This can be seen as an extension to our Sequence Strategy with the only difference that the sequences have to be taken not only from the current document but from the cache memory and all multi-word proper names and lowercased word bigrams identified in a document are to be appended to that cache. This strategy covered another 2% of unresolved cases with accuracy of above 99%.
Table 4
Cascading capitalized words disambiguation strategies.

| Strategy          | MUC'7 Corpus | Brown Corpus |
|-------------------|--------------|--------------|
|                   | Proper | Common | All  | Proper | Common | All  |
| **Sequence Strategy** |       |        |      |       |        |      |
| coverage          | 9.3%   | 16.7%  | 14.2% | 8.5%   | 25.5%  | 22.5% |
| accuracy          | 98.7%  | 100%   | 99.7% | 99.2%  | 100%   | 99.8% |
| **Stop List**     |       |        |      |       |        |      |
| coverage          | 6.1%   | 69.3%  | 49.5% | 6.8%   | 62.5%  | 52.9% |
| accuracy          | 100%   | 100%   | 100%  | 99.8%  | 99.9%  | 99.9% |
| **Single Word**   |       |        |      |       |        |      |
| coverage          | 50.8%  | 3.4%   | 18.3% | 35.9%  | 4.6%   | 9.9%  |
| accuracy          | 99.8%  | 97%    | 99.4% | 99.2%  | 97.6%  | 98.6% |
| **After Abbr**    |       |        |      |       |        |      |
| coverage          | 10.5%  | 0.4%   | 3.4%  | 29.7%  | 0.2%   | 5.25% |
| accuracy          | 98.8%  | 100%   | 99.6% | 99.2%  | 95.3%  | 99.1% |
| **Quotes/Brackets** |      |        |      |       |        |      |
| coverage          | 6.8%   | —      | 2.0%  | 6.2%   | —      | 1.0%  |
| accuracy          | 98.2%  | —      | 98.2% | 98.1%  | —      | 98.1% |
| **TOTAL**         |       |        |      |       |        |      |
| coverage          | 82.6%  | 89.9%  | 87.6% | 87.7%  | 92.5%  | 91.7% |
| accuracy          | 99.4%  | 99.9%  | 99.85%| 99.2%  | 99.8%  | 99.7% |

6 Assigning Sentence Breaks

Now after we have identified most of the abbreviations and more importantly most of the ordinary words in the text, and after we have classified the words into proper nouns and common words, we can carry out assignments of potential sentence breaks. First, we assign as fullstops \( T = ', \) potential sentence breaking punctuation which follow an ordinary word and is followed by a capitalized word. About 92% of potential sentence boundaries of the Brown Corpus and 83% of those in MUC'7 Corpus fall in this category.

We can also easily disambiguate most of the periods which follow abbreviations and potential abbreviations. A period is sentence terminal when it is preceded by an abbreviation or potential abbreviation and

- is not followed by any other token apart from closing brackets, quotes and sentence closing punctuation (., ?). This takes care of cases like “Ltd."/P”. Note that we eliminated such easy cases from the Brown Corpus as described in section 3.3;

- is followed by optional closing brackets and quotes and then by a sentence closing punctuation (., ?). This covers cases like “Ltd.. Boeing” or “Ltd.). Boeing”. Note that we eliminated such easy cases from the Brown Corpus as described in section 3.3;

- is followed by a capitalized word which has been disambiguated as a common word by the methods described in section 5. We also allow for optional opening and closing brackets or quotes between the period and capitalized common word. Examples of contexts handled by this rule are “Ltd. The”, “Ltd.) The” or “Ltd.) (The”;

24
Table 5
Disambiguated sentence breaking punctuation when attempted by the full system and the system without abbreviation lexicon. The two “total” results correspond to the rule-based assignments (Total) and rule-based assignments complimented with the default strategy to assign a period as abbreviation internal in unresolved cases (TOTAL).

|                      | Example | Full-stop | Abbr. | Both | Total | TOTAL |
|----------------------|---------|-----------|-------|------|-------|-------|
| Brown Corpus         |         |           |       |      |       |       |
| Proportion          | Mr. Brown | 92.5% | 7.1% | 0.4% |       |       |
| Full System          | Accuracy | 100% | 98.6% | 96.2% | 99.9% | 99.7% |
|                      | Coverage | 99.9% | 95.8% | 67.4% | 99.4% | 100%  |
| System with no       | Accuracy | 99.8% | 99.0% | 91.7% | 99.75% | 99.55%|
| Abbr. Lex.           | Coverage | 99.9% | 91.5% | 41.8% | 99.0% | 100%  |
| MUC’7 Corpus         |         |           |       |      |       |       |
| Proportion          |           | 83.2% | 15.2% | 16%  |       |       |
| Full System          | Accuracy | 100% | 99.7% | 100% | 99.9% | 99.3% |
|                      | Coverage | 99.3% | 85.9% | 95.4% | 97.2% | 100%  |
| System with no       | Accuracy | 98.9% | 99.0% | 100% | 99.8% | 99.2% |
| Abbr. Lex.           | Coverage | 99.9% | 82.8% | 63.6% | 96.7% | 100%  |

A period is part of abbreviations (T=’A’) when it is preceded by a word which has been recognised as abbreviation and

- is followed by a comma, lowercased word, or a number. There can be optional quotes and brackets in between them. Here are some examples: “Ltd. said”, “Ltd., and”, “Ltd.] and”;
- this abbreviation is in the first five tokens of the sentence but not the first one. For instance, in the sentence

“At 3 p.m. Continental signed the deal.”

the abbreviation “p.m.” is not sentence terminal since it is in the first five tokens of this sentence;
- this abbreviation is the first token of the sentence and is not followed by a frequent sentence starter. Here we take care of situations when a single letter can be a list iterator e.g. “A. This is”, or some abbreviations like “P.S.” or “BTW.” are usually used before a sentence e.g. “P.S. Don’t forget…”;
- this abbreviation is capitalized and the following capitalized word has been disambiguated as a proper name by the method described in section 5.4. This is a very productive rule which takes care of cases like “Sen. Brown” and although one can imagine cases like “Ltd. Brown” when the word “Brown” starts a new sentence the empirical study confirmed that this happens very infrequently and the accuracy (precision) of this rule was measured at about 99.5%;

Table 6 displays the results of the applications of the above mentioned rules to our two corpora. We tested these rules in conjunction with two configurations for the abbreviation identification: one is the full fledged configuration which includes the list of 270 abbreviations, the surface guessing heuristics and the positional guessing strategy and the other uses only the guessing heuristics and the positional strategy.

A period is part of abbreviations (T=’A’) when it is preceded by a word which has been recognised as abbreviation and

- is followed by a comma, lowercased word, or a number. There can be optional quotes and brackets in between them. Here are some examples: “Ltd. said”, “Ltd., and”, “Ltd.] and”;
- this abbreviation is in the first five tokens of the sentence but not the first one. For instance, in the sentence

“At 3 p.m. Continental signed the deal.”

the abbreviation “p.m.” is not sentence terminal since it is in the first five tokens of this sentence;
- this abbreviation is the first token of the sentence and is not followed by a frequent sentence starter. Here we take care of situations when a single letter can be a list iterator e.g. “A. This is”, or some abbreviations like “P.S.” or “BTW.” are usually used before a sentence e.g. “P.S. Don’t forget…”;
- this abbreviation is capitalized and the following capitalized word has been disambiguated as a proper name by the method described in section 5.4. This is a very productive rule which takes care of cases like “Sen. Brown” and although one can imagine cases like “Ltd. Brown” when the word “Brown” starts a new sentence the empirical study confirmed that this happens very infrequently and the accuracy (precision) of this rule was measured at about 99.5%;

Table 6 displays the results of the applications of the above mentioned rules to our two corpora. We tested these rules in conjunction with two configurations for the abbreviation identification: one is the full fledged configuration which includes the list of 270 abbreviations, the surface guessing heuristics and the positional guessing strategy and the other uses only the guessing heuristics and the positional strategy.
Although there are fewer truly ambiguous sentence breaks in MUC’7 Corpus (5%) than in the Brown Corpus (6%), MUC’7 Corpus presented a bigger challenge to our system in terms of coverage. For both our corpora the fully equipped system in its assignments achieved very high accuracy of 99.9% but it failed to make a decision to 0.6% of the potential sentence breaks in the Brown Corpus and to 2.8% of the potential sentence breaks in MUC’7 Corpus. This difference is due to the periods which are sentence internal i.e. parts of abbreviations. In the Brown Corpus the system managed to disambiguate 95.8% of such cases whereas in MUC’7 Corpus only 85.9%, and at the same time the proportion of such abbreviations was twice as high in MUC’7 Corpus. At the same time we see a better handling of periods which belong to abbreviations and are sentence terminal at the same time on MUC’7 Corpus. This can be explained by the fact that we deleted paragraph marks and double periods in the Brown Corpus as explained in section 3.3.

To handle the unassigned cases we employed a default strategy which classifies a period as abbreviation internal (i.e. not a sentence break) in the unresolved cases. Using this strategy we achieved 100% coverage but with about 0.2% drop in accuracy on the Brown Corpus and 0.6% drop in accuracy on MUC’7 Corpus. This comes from the fact that a larger proportion of ordinary words (+0.6%) was recognized in the Brown Corpus. The unrecognized ordinary words were treated as abbreviations by the default strategy. Thus although on the Brown Corpus the default strategy produced 47% error rate whereas on MUC’7 Corpus only 22% error rate, the larger proportion of unresolved cases of MUC’7 Corpus ensured 0.4% larger absolute drop in accuracy.

To our great satisfaction the system also demonstrated a strong performance even without using the abbreviation list. The abridged system coverage was only 0.4% lower than that of the full system and the accuracy was lower only by 0.1%. The only significant performance difference was detected on sentence terminal abbreviations. They, however, were too infrequent to make the difference in general. When we applied the default assignments to the unresolved by the abridged system cases, the overall performance was only 0.1-0.15% lower than that of the full system. In terms of absolute numbers the configuration without the abbreviation list achieved 99.55% accuracy on the Brown Corpus and 99.2% on MUC’7 Corpus with 100% coverage.

7 Further extensions

7.1 Handling Russian News

To test the generality of our approach we applied it to a corpus of BBC news in Russian. We collected this corpus from the Internet http://news.bbc.co.uk/hi/russian/world/default.htm over a period of 30 days. These gave us a corpus of 300 short documents - one two paragraphs each. We built the supporting resources (the lists of frequent starting words and frequent proper names) from 364,000 Russian corpus of European Corpus Initiative (ECI). We did not equip the system with the lexicon of abbreviations, though.

Since unlike English, Russian is a highly inflected language, we had to deal with the case normalization issue. Before applying our method, we converted each word in the text to its main form - singular nominative case for nouns, infinitive for verbs, etc. - by means of a Russian morphological processor described in (Mikheev&Liubushkina, 1995). For words which could be normalised to several main forms, i.e. when secondary forms of different words coincide, we retained all the main forms. Since the documents in our BBC news corpus were rather short, we implemented a word cache module as described in section 5.5. This allowed us to reuse information across the documents.

Russian proved to be a simpler case than English for the sentence boundary and capitalized word disambiguation tasks. Firstly, on average Russian words are longer than
English words, thus the identification of abbreviations is simpler. Secondly, proper names in Russian much less frequently coincide with common words – this makes their disambiguation in mandatory positions easier. The overall performance reached 99.9% accuracy on sentence boundaries and 99.7% accuracy on ambiguously capitalized words with coverage on both tasks at 100%.

7.2 Building a POS Tagger

We also performed experiments in casting our approach into a consistent probabilistic framework. Instead of applying our heuristics for assigning proper names and sentence boundaries, we trained a statistical model which utilized the information which is normally supplied to our heuristics. This information includes whether a word was seen lowercased and/or capitalized in the same document, whether a sequence of capitalized or lowercased words was seen in the same document, whether a word is in the list of frequent starting words and/or in the list of frequent proper names, etc.

As for the probabilistic framework, we had a choice of training either a probabilistic classifier which uses a limited context akin to those described in section 2.1.3 or train a part-of-speech tagger which applies path probability maximization to a larger context (section 2.2.2). We decided that the tagging model is more suitable for our task because many disambiguation clues depend on the distribution of the syntactic categories of words in the surrounding context. It is naturally true for proper name disambiguation but it is also true for sentence boundary disambiguation.

One of the most difficult cases in sentence boundary disambiguation is when an abbreviation is followed by a proper name: sometimes they belong to different sentences (example 3.3 and example 3.4) and sometimes they are in the same sentence (example 3.1 and example 3.2). However, syntactic context can disambiguate many of such cases. For instance, when a noun group with an abbreviation is following a verb, and the next proper name is followed by a verb – they clearly belong to different clauses and hence to different sentences (example 3.3 and example 3.4).

Part-of-speech categories of words in the local context were also used by (Palmer & Hearst, 1997) in their SATZ system (section 2.1.3). However, these syntactic categories were assigned on the basis of the most frequent part-of-speech tag for an ambiguous word i.e. using only unigrams. The unigram approach is known to perform at about 90% accuracy while the bigram and trigram approaches which are used in most of POS taggers reach 96%-97% accuracy. (Palmer & Hearst, 1997) found difficulty in applying a POS tagger instead of the unigram assignments: “However, requiring a single part-of-speech assignment for each word introduces a processing circularity: because most part-of-speech tags require predetermined sentence boundaries, the boundary disambiguation must be done before tagging. But if the disambiguations done before tagging, no part-of-speech assignments are available for the boundary determination system”. Thus they opted for the unigram model.

This circularity, however, can be overcome. In section 4 we have shown that in mixed-case texts it is possible to reliably identify abbreviations and non-abbreviations and at the same time many abbreviations are not ambiguous as sentence breaks. This allows for reliable assignment of 94-95% of potential sentence boundary punctuation. Thus only about 5-6% of the potential sentence boundary punctuation are ambiguous. Such punctuation can take three possible tags – T = ‘.’ - a fullstop, T = ‘A’ - part of abbreviation, T = ‘A’ + ‘.’ - part of abbreviation and fullstop at the same time. This is exactly the task for a POS tagger. Thus we suggest to tokenize periods (and other potential sentence breaking punctuation) as separate tokens as described in section 3.3, and apply the standard POS tagging technique to the punctuation which in our treatment is ambiguous, contrary to the model adopted in Penn Treebank (Marcus et al., 1993). The technical issue of
having too long sequences for the optimal path search can be overcome by breaking these sequences at unambiguous words. For instance, for a bigram tagger a sequence of words can be terminated at an unambiguous word; for a trigram tagger a sequence can be terminated when two unambiguous words follow each other, etc.

It is difficult to incorporate features from non-local context into traditional Hidden Markov Model taggers. Transformation rules of Brill probably are capable to utilize non-local features better. Arguably the most consistent framework to bring together diverse information sources is Maximum Entropy (ME) modelling (Ratnaparkhi, 1996) developed a ME tagger but this tagger did not use the full strength of the underlying model because the features in use were generated only from the local context. We built an ME tagger with a similar set of features as (Ratnaparkhi, 1996) and also using our non-local features of word distribution in the entire document. Another, difference to (Ratnaparkhi, 1996) work is that we used a feature collocation lattice (Mikheev, 1998) as a feature discovery mechanism in our implementation of the ME tagger.

Our extra features were used only with the word before and the word after a potential sentence breaking punctuation. Here is an example of the feature vectors for the context “Mr. Brown” when the word “Brown” was seen in the document capitalized in an unambiguous position:

| Word | Feature                  | Value       | Value       | Value       |
|------|--------------------------|-------------|-------------|-------------|
| -2   | lexical                  | "Mr"       | "Mr"       | "Mr"       |
|      | POS tag                  | NNP         | NNP         | NNP         |
|      | abbreviation             | YES         | YES         | YES         |
|      | possible abbreviation    | YES         | YES         | YES         |
| -1   | lexical                  | "."        | "."        | "."        |
|      | POS tag                  | A           | *           | .           |
| 0    | lexical                  | "Brown"    | "Brown"    | "Brown"    |
|      | POS tag                  | NNP         | JJ          | NNP         |
|      | length of seen capitalized sequence | 1 | 1 | 1 |
|      | length of seen non-capitalized sequence | 0 | 0 | 0 |
|      | is frequent sentence starter | NO | NO | NO |
|      | is frequent proper name   | YES         | YES         | YES         |

Here we see features generated for three (out of six) different possible paths over “Mr. Brown”. The -2 word (“Mr”) is unambiguous and its values are identical for all three different paths. It has specific to our approach extra features “abbreviation” and “possible abbreviation” produced as explained in section 4. The next word -1 is a period and it is ambiguous among three possible tags (A, * and .). The word in focus (0) is the word “Brown” for which we supply the features “length of seen capitalized sequence” and “length of seen non-capitalized sequence”. These features are generated similar to the Sequence Strategy (section 5.1) and to the Single Word Assignment (section 5.3) and record the length of unambiguously seen elsewhere in the document capitalized or non-capitalized sequences of words which can be matched from the current position. In our particular case the word “Brown” was not seen with the word which follows it (not shown in our example) but it was seen unambiguously capitalized and was not seen lowercased in the current document. The features, “is frequent sentence starter” and “is frequent proper name” come from our two lists (section 3.2) and akin to the Frequent List Lookup Strategy (section 5.2.)

Unlike our original method, here we don’t need to prioritize the information sources over each other – the tagger learns this itself. We trained this tagger on Wall Street Journal and applied it to our two corpora. As we expected the performance improved: the tagger assigned proper names with accuracy 99.6% (coverage 100%) on both our
corpora and it assigned sentence boundaries with accuracy 99.87% on the Brown Corpus and 99.5% on MUC'7 Corpus. These are rather minor improvements (about 0.2%) but they come from resolving difficult cases and in fact they reduce the error rate by a third. For instance, the tagger correctly assigned most of the honorific titles as proper names – and this was one of the problems for the Single Word Assignment. It looks like the tagger learned the rule to choose NNP over NN when the next word is capitalized. The tagger also correctly disambiguated more abbreviations at the end of sentence. There we had a very low coverage (about 60%) which was translated into 40% error rate after applying the default assignment. The tagger managed to achieve only 10% error rate on this category. However, despite these obvious improvements, such cases were rather infrequent in our both corpora (under 1.5%) and thus did not contribute to a larger improvement to the overall performance.

8 Discussion

In this paper we presented an approach which tackles three problems: sentence boundary disambiguation, disambiguation of capitalized words when they are used in positions where capitalization is expected and identification of abbreviations. All these tasks are important tasks of text normalization, which is a necessary phase in almost all text processing activities. As opposed to most of the existing approaches which use primarily local contextual information, we decided to put the emphasis on the information which is drawn from the entire document.

The main feature of our approach is that it uses a minimum of pre-built resources – we use only a list of common words and lists of the most frequent words which appear in the sentence-stating positions. These lists can be acquired without any human intervention as we have shown in section 3.2. To compensate for the lack of pre-acquired knowledge, the system tries to dynamically infer disambiguation clues from the entire document itself. This makes our approach domain independent and closely targeted to each document.

Our approach is more akin to that implicitly adopted in Information Retrieval rather than to those of Natural Language Processing. The task of Information Retrieval is to categorize (or index) a document as a function of words and phrases which constitute this document. Since important words normally are used in a document more than once and also in many different contexts, usually some of these contexts are less ambiguous than others and thus allow for the robust identification of task-important characteristics of these words. Therefore it is enough to use only the unambiguous occurrences of important words, or in other words not to invest into the disambiguation of ambiguous usages. In Natural Language Processing the task is to identify the characteristics for each individual occurrence of a word. In most cases the processing unit in NLP is a sentence but more realistically only a limited local context of a word is taken into account in each particular word occurrence.

In our approach we concerned with assigning each individual occurrence of interesting for us words, but at the same time we don’t try to resolve each of such occurrence individually applying its local context. Instead we are scanning the entire documents for the contexts where the words in question are used unambiguously and this gives us grounds to resolve ambiguous contexts. For instance, for the disambiguation of capitalized words in mandatory positions the above reasoning can be crudely summarized as follows: if we detect that a word has been used capitalized in an unambiguous context (not in a mandatory position), this increases the chances for this word to act as a proper name in mandatory positions in the same document. And conversely if a word is seen only lowercased, this increases the chances to downcase it in mandatory positions of the same document. This approach managed to classify with almost perfect accuracy nine words
out of ten leaving one word unassigned. The simplest default assignment strategy of using the list lookup approach applied to the 10% of unresolved cases reduced the overall accuracy to about 99% but allowed to cover all ambiguously capitalized words.

The same logic applies to abbreviations. Although a short word which is followed by a period is a potential abbreviation, the same word when occurred in the same document in a different context can be unambiguously classified as an ordinary word if it is used without trailing period, or it can be unambiguously classified as an abbreviation if it is used with a trailing period and is followed by a lowercased word, a number or a comma. This in conjunction with the surface guessing heuristics (section 4) allowed our system to produce strong performance even without a precompiled list of abbreviations.

Our treatment of the sentence boundary problem is also different from the approaches proposed before. Instead of solving sentence boundary problem directly, we invested in the identification of abbreviations and the disambiguation of capitalized words. The information from these two sources largely solves the sentence boundary problem leaving only about 5% unresolved. To solve these cases we applied only three very general heuristics which accurately covered most of the ambiguous cases. The performance of the system reached 99.9% accuracy with coverage of 99.4% on the Brown Corpus and with coverage of 97.2% on MUC'7 Corpus. When we applied a default rule to classify all unassigned cases as sentence internal periods we covered all potential sentence boundaries but with some drop in accuracy: 99.7% on the Brown Corpus and 99.3% on MUC'7 Corpus.

We deliberately shaped our approach that it does not rely on pre-compiled statistics but rather acts by analogy. This is because the most interesting events are inherently infrequent and, hence, are difficult to collect reliable statistics for, and at the same time pre-compiled statistics would be smoothed across multiple documents rather than targeted to a specific document. Another reason is that the use of a statistical model would require a set of pre-labelled examples and thus additional labour investments. Instead of using a statistical model we used a very small number of heuristics which utilize the features which we collect from the entire document. Despite of the simplicity, the performance of the system was higher than previously reported. Its proper name disambiguation error-rate was about 1% whereas the error rate of a POS tagger would be about 3-4%. The error rate on sentence boundaries in the Brown Corpus reached the lowest quoted before (Riley, 1989), but our system worked on a more difficult version of the Brown Corpus where paragraph information and double periods were stripped off (section 3.3). Unlike (Riley, 1989), our system also did not use any corpus and language specific information such as characteristics and probabilities for individual words and abbreviations.

The main success of our approach is in the nature of features in use. When we casted our features into a proper probabilistic framework (a POS tagger) we measured a further improvement on both tasks. The tagger achieved the accuracy of 99.87% on sentence breaks on the Brown Corpus which is the highest quoted so far. The tagger also improved about 0.6% on our original method scoring 99.6% accuracy on ambiguously capitalized words. This basically shows that of course a proper probabilistic framework is more accurate than a bunch of heuristics but the main advantage is in the utilized information. The development of a statistical model, on the other hand, requires additional investments in terms of labour and processing speed which may not be acceptable for some applications.

We also investigated the portability of our approach to other languages and obtained very encouraging results on a corpus of news in Russian. This strongly suggests that this method is adequate for the majority of European languages since they share the same principles of capitalization and word abbreviation. Our method, however, in the form described in this paper is suitable only to mixed-case texts since it uses capitalization as an
Andrei Mikheev

Periods, Capitalized Words, etc.

important feature. Method of (Palmer&Hearst, 1997) for sentence boundary disambiguation does not rely on capitalization and produces similar results on mixed-case texts and on single case texts. However, the results obtained with our method on mixed-case texts are much higher than those of (Palmer&Hearst, 1997) which proves that capitalization is an important feature and one should not neglect it when it is available.

Acknowledgments

The work reported in this paper was supported in part by grant GR/L21952 (Text Tokenisation Tool) from the Engineering and Physical Sciences Research Council, UK, and also it benefitted from the ongoing efforts in building domain independent text processing software at Harlequin Ltd.

References

J. Aberdeen, J. Burger, D. Day, L. Hirschman, P. Robinson and M. Vilain. 1995. MITRE: Description of the Alembic system used for MUC-6. In Proceedings of the Sixth Message Understanding Conference (MUC-6) Columbia, Maryland, November 1995
B. Baldwin, C. Doran, J. Reynar, M. Nir, B. Srinivas and M. Wesson. 1997. EAGLE: An Extensible Architecture for General Linguistic Engineering In Proceedings of RIAO '97, Montreal, June 1997
Burnage, G. 1990. CELEX: A Guide for Users. Nijmegen: Centre for Lexical Information.
E. Brill. 1995. Transformation-based error-driven learning and natural language parsing: a case study in part-of-speech tagging. In Computational Linguistics 21 (4), pp. 545-565 The MIT Press.
L.L. Cherry and W. Vesterman. 1991. Writing tools – the STYLE and DICTION programs. In 4.3 BSD UNIX System Documentation. University of California, Berkeley
N. Chinchor. 1998. Overview of MUC-7. In Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference held in Fairfax, VA, April 29-May 1, 1998. www.muc.sai.columbia.edu/muc7/proceedings/overview.html
K. Church. 1995. One Term Or Two? In Proceedings of the 18th Annual Internationals ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95), Seattle
K. Church. 1988. A Stochastic parts program and noun-phrase parser for unrestricted text. In Proceedings of the Second ACL Conference on Applied Natural Language Processing (ANLP'88), Austin, Texas
W. Francis and H. Kucera 1982 Frequency Analysis of English Usage. Boston MA: Houghton Mifflin.
P. Hayes. 1994. NameFinder - Software that finds names in text. In Proceedings of RIAO'94 pp 762-774. New York.
T.L. Humphrey and F. Zhou. 1989. Period disambiguation using a neural network In IJCNN: International Joint Conference on Neural Networks, pp. 606 Washington, D.C.
J. Kupiec. 1992. Robust Part-of-Speech Tagging Using a Hidden Markov Model. In Computer Speech and Language
M.E. Lesk and E.Schmidt. 1975. LEX - a lexical analyser generator Computing Science Technical Report 39. AT&T Bell Laboratories, Murray Hill, N.J.
M.Y. Liberman and K.W. Church. 1992. Text analysis and word pronunciation in text-to-speech synthesis. In S. Furui and M. Sondhi, eds., Advances in Speech Signal Processing, Marcel Dekker, Inc., New York
I. Mani and T.R. MacMillan. 1995. Identifying Unknown Proper Names in Newswire Text In B. Boguraev and J. Pustejovsky, eds., Corpus Processing for Lexical Acquisition, MIT Press.
M. Marcus, M.A. Marcinkiewicz, and B. Santorini. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. In Computational Linguistics, vol 19 (2), ACL.
A. Mikheev. 1997 (a). Automatic Rule Induction for Unknown Word Guessing. In Computational Linguistics 23 (3), pp. 405-423
A. Mikheev. 1997 (b). LT POS – the LTG part of speech tagger. Language Technology Group, University of Edinburgh. www.ltg.ed.ac.uk/software/pos
A. Mikheev. 1998. Feature Lattices for Maximum Entropy Modelling. In Proceedings of the 36th Conference of the Association for Computational Linguistics (ACL/COLING'98), pp 848-854. Montreal, Quebec.
A. Mikheev. 1999. A Knowledge-free Method for Capitalized Word Disambiguation. In *Proceedings of the 37th Conference of the Association for Computational Linguistics (ACL’99)*, pp. 159-168. University of Maryland, USA.

A. Mikheev and L. Liubushkina. 1995. Russian morphology: An engineering approach. In *Journal for Natural Language Engineering*, vol 1 (3), pp.235-260. Cambridge University Press.

A. Mikheev, C. Grover and M. Moens. 1998 (a). Description of the LTG system used for MUC-7. In *Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference held in Fairfax, VA, April 29–May 1, 1998.*

http://www.muc.saic.com/muc_7_proceedings/ltgmuc7.ps

A. Mikheev, C. Grover and C. Matheson. 1998 (b). TTT: Text Tokenisation Tool. Language Technology Group, University of Edinburgh.

http://www.ltg.ed.ac.uk/software/TTT/index.html

H. Muller, V. Amerl and G. Natalis. 1980. Worterkennungsv erfahren als Grundlage einer Universalmethode zur automatischen Segmentierung von Texten in Satze. Ein Verfahren zur maschinellen Satzgrenzenbestimmung im Englischen In *Sprache und Datenverarbeitung*, 1

G.T. Nicol. 1993. Flex -The Lexical Scanner Generator. The Free Software Foundation. Cambridge, MA.

R.J. Quillan. 1996. ch.5: Programs for Machine Learning. Morgan Kaufman, San Mateo, CA.

D. D. Palmer and M. A. Hearst. 1997. Adaptive Multilingual Sentence Boundary Disambiguation. In *Computational Linguistics*, 23 (2), pp. 241-269

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pp. 133-142 University of Pennsylvania.

J. C. Reynar and A. Ratnaparkhi. 1997. A Maximum Entropy Approach to Identifying Sentence Boundaries. In *Proceedings of the Fifth ACL Conference on Applied Natural Language Processing (ANLP’97)*, Washington D.C., ACL.

M.D. Riley. 1989. Some applications of tree-based modelling to speech and language indexing. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pp 339-352 Morgan Kaufman.

R.J. Schwartz (editor). 1955. The Complete Dictionary of Abbreviations. Thomas Y. Crowell Company: New York, 1955.

Berkeley University. 1999. Dictionary of Abbreviations and Acronyms in Geographic Information Systems, Cartography, and Remote Sensing

http://www.lib.berkeley.edu/EART/abbrev.html