Automatic disabbreviation by using context information

Terada Akira and Tokunaga Takenobu

Department of Computer Science
Tokyo Institute of Technology
{aterada,take}@cl.cs.titech.ac.jp

Abstract Unknown words such as proper nouns, abbreviations, and acronyms are a major obstacle in text processing. In particular, abbreviations are often used in specific domains. In this paper, we propose an automatic disabbreviation method using context information. In past research, a dictionary has conventionally been used to search abbreviation expansion candidates for an abbreviation. We use an abbreviation-poor text of the same domain instead of a dictionary. We calculate the plausibility of expansion candidates based on the similarity between the context of a target abbreviation and that of its expansion candidates. The similarity is calculated using the vector space model, in which each vector element consists of surrounding words. Experiments using about 10,000 documents in the aviation domain showed that the proposed method is superior to past methods by 10% in precision.

Keywords : *unknown words, abbreviation, context information*

文脈情報を使用した略語の自動復元

寺田 昭　　徳永 健伸

東京工業大学 大学院情報理工学研究科
{aterada,take}@cl.cs.titech.ac.jp

あらまし テキスト処理において、固有名詞、略語、頭字語などの未知語の処理は難しい問題である。未知語は、情報検索、テキストデータマイニングなどの応用システムや人間の理解に悪影響を与える。特に特定の対象領域に関するテキストでは略語は多用される．本論文では、略語の自動復元について述べる。従来の研究では、略語の復元の為の候補語の選定に辞書を使用していたが、本論文では、同じドメインのテキストで略語をできるだけ含まないようなものを知識源として使用する。候補語の中から正しい復元形を見つけるために、復元対象となる略語の文脈情報と知識源に含まれる復元形の候補語の文脈情報の類似性を使用した。文脈情報とは、語の前後に出現する単語を意味する。ベクトル空間法において、略語および候補語の近似度を文脈情報により計算し、候補語の中から正しい復元形を選定した。航空関係の10,000文書を対象に実験をおこなったところ従来法に比較して精度で約10%の改善を得た．

キーワード：未知語, 略語, 文脈情報

1 Introduction

Unknown words degrade the performance of text processing applications, such as information retrieval and text data mining. By unknown words, we mean tokens that are not recognized by the system, including numbers, proper nouns, abbreviations, acronyms, misspellings, run-on words and so on. In this paper, we focus on abbreviatons, and propose a method to decipher them. Past disabbreviation methods have tended to return multiple candidate words and force the user to make the final choice, due to low system precision. When applying the system to a task, such as text-to-speech synthesis, however, the system is required to perform fully automatic disabbreviation.

Most existing disabbreviation techniques use only the abbreviations themselves, without taking into account any information that might be gleaned from the linguistic or textual context in which the string appears. When people expand the abbreviation to the original word, they rely heavily on context information around the abbreviation [RL95]. To improve the system performance, context information is necessary.

In a real world, abbreviations are used in various domains and in various ways. To expand abbreviations, domain specific knowledge and dictionaries seem to be indispensable. Instead of constructing domain specific knowledge and dictionaries, we use the same domain corpus with less abbreviations.

The aviation domain involves the frequent use of abbreviations. We use ASRS (Aviation Safety Reporting System) Dataset[1], in which abbreviations are included in about 11.0% and acronyms 2.0% of all word occurrences based on manual investigation of an 8,000 word sample. The ASRS Dataset is a collection of reports submitted by pilots, air traffic controllers, flight attendants, mechanics, ground personnel, and others involved in or observers of an incident or situation in which aviation safety is compromised. Incident reports are read by at least two ASRS aviation safety analysts. This report system is designed to maintain confidentiality and the anonymity of the reporter, so all information which may identify the reporter has been deleted including the aircraft name. In this manner, an unspecified name "X", such as "aircraft X" is used very often. These unspecified names are also unknown words. Our task is also to distinguish these unspecified names and abbreviations. The ASRS Dataset is structured and includes the report number, local date, weather conditions as well as text data, which is divided into a narrative and synopsis. The synopsis is shorter than the narrative and is a summary of the narrative. We used the narrative portion of the ASRS Dataset about 38 words per document in our experiment.

We define abbreviations to be a short representation of a single word, e.g. "TKOF - takeoff", and acronyms to be a short representation of more than one word, e.g. "ILS - Instrument Landing System". Hereafter, we denote abbreviations in upper case, and the original word form in lower case.

In the surface, disabbreviation appears similar to spelling error correction [Kuk92b], but as [RL95] pointed out, abbreviations are much shorter than their original word forms and lose more information than misspellings.

[Kuk92b] reported that most nonword misspellings tend to be within two characters of the correct spelling. In our experiment, we found abbreviations to be 4.3 characters shorter than their original word form. This means that it is difficult to decipher the original form of the abbreviation with only character information.

The disabbreviation task is divided into three sub-tasks; detection, expansion and ranking. Abbreviation detection is the easiest of these sub-tasks. [RL95, Too00] used a dictionary to search for abbreviation expansion candidates. In their method, if a general dictionary is used, many irrelevant abbreviation expansion candidates will be produced, which might degrade the system performance.

To search for abbreviation expansion candidates, we use an abbreviation-poor corpus of the same domain. An abbreviation-rich corpus includes a lot of abbreviations and an abbreviation-poor corpus does not include many abbreviations. To rank them effectively and efficiently, we propose a disabbreviation method which primarily uses context information, i.e. words appearing before and after the abbreviation. Using an abbreviation-poor corpus, we can collect words around abbreviation expansion candidates. We assume that the words appearing around the abbreviation and those appearing around its original word form are statistically similar. The idea is that if we use an abbreviation-poor corpus, we can get abbreviation expansion candidates which appear in an abbreviation-rich corpus, without using a dictionary. There are two reasons why we use an abbreviation-poor corpus: (1) to use word frequency information, and (2) an abbreviation-rich corpus may not include the original form of a given abbreviation. For an abbreviation-rich and an abbreviation-poor corpus belonging to the same domain, we can expect that most of the original word forms of abbreviations in the abbreviation-rich corpus will be included in the abbreviation-poor corpus, share common context.

For this purpose, we use the NTSB (National Transportation Safety Board)[2] aviation accidents synopses as our abbreviation-poor corpus. "TKOFF" appears 1,156 times in the ASRS dataset, but "takeoff" never appears, whereas "TKOFF" appears 28 times and "takeoff" 344 times in the NTSB dataset ("TKOFF" is an abbreviation of "takeoff" in the ASRS dataset). Characteristics of the ASRS and NTSB dataset are given in Table 1.

We made the following assumptions about abbreviations:

1. Abbreviations include less characters than their original word forms.

2. The characters used in an abbreviation are a subset of those appearing in the original word form and

[1] ASRS data is available at http://asrs.arc.nasa.gov/

[2] URL:http://www.ntsb.gov/aviation/aviation.htm

Table 1: ASRS and NTSB datasets

	ASRS	NTSB
#docs	2,648	3,937
#words (token)	677,725	426,717
#words (type)	18,422	14,940
ave. #words/doc	260	108
#unkown words (token)	100,983	33,575
#unkown words (type)	6,077	7,190
ave. #unkown words/doc	38.1	8.5
unkown word ratio	14.9%	8.5%

occur in the same order, except “X”, “–”, and “/”, e.g. “TKOF – takeoff”, “WX – weather”.

3. Abbreviations are not real words.
4. Abbreviations correspond to unique words.

One may think that if a fixed abbreviation list is available, no problem exists. However in a real world, different persons or even the same person may abbreviate the same word differently, so a fixed list of abbreviations has limited application (e.g., both “A/C” and “ACFT” stand for “aircraft”).

2 System design

We consider unknown words as abbreviation candidates (hypothesis 3). To detect unknown words, we take a tagged corpus and make use of “unknown word” information annotated by the tagger. When an unknown word appears, it may or may not be an abbreviation. The system searches for abbreviation expansion candidates from an abbreviation-poor corpus using hypotheses 1 and 2. Then the system collects the words around the abbreviation candidate and abbreviation expansion candidates. Using this information, the system decides which abbreviation expansion candidate is most appropriate for the original form of the abbreviation or alternatively that none of the abbreviation expansion candidates are appropriate for the original form of the abbreviation, that is, the system decides that the unknown word is not an abbreviation.

As is evident above, our system can be divided into three parts: an unknown word detection, abbreviation candidate expansion, and ranking.

We tagged the ASRS and NTSB datasets using TreeTagger [Sch94, Sch95]. TreeTagger annotates text with part-of-speech and lemma information. TreeTagger annotates words as “unknown” when they are not listed in the tagger dictionary.

We pick “unknown” words as abbreviation candidates only when they are annotated with the Noun, Verb, Adjective, Adverb or Preposition part-of-speech tags (POS filter).

Next, the words which appear before and after the abbreviation candidate are collected. As TreeTagger provides the root form of each word, we use this information in word collection, and no further stemming is required. In collecting the words around the abbreviation candidate, all occurrences of the same type of the abbreviation are collected together (hypothesis 4). We used a window of size 3. In our preliminary test, we experimented windows of size 3 and 5. We found no noticeable difference between them. The window size may be greater than the syntactical scope of the word, but this is beyond the scope of this paper.

Next, abbreviation expansion candidates for the abbreviation are searched for using hypotheses 1 and 2 in an abbreviation-poor corpus. We take only Nouns, Verbs, Adjectives, Adverbs and Prepositions as abbreviation expansion candidates (POS filter). The same window size was used here as for the abbreviation context. Figure 1 illustrates an overview of the system, and Figure 2 shows an example of content information.

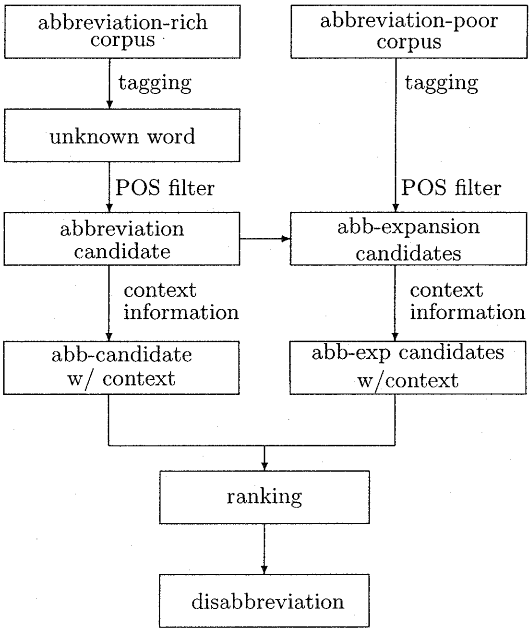

Figure 1: System overview

2.1 Term weighting

We take the word context around the abbreviation to be a query and each expansion candidate to be a document and the word context around the expansion candidate to be the content of the document, in information retrieval terms.

In information retrieval, many of the most frequently occurring words such as “the” and “or” are eliminated from indexing terms, because these words have no specific meaning in each individual document [FBY92]. However in our case, many abbreviations are surrounded by function words representative of the nature of the abbreviation, as an abbreviation is not a document, but a single word. As such, we did not eliminate function words. A preliminary test supported this design choice.

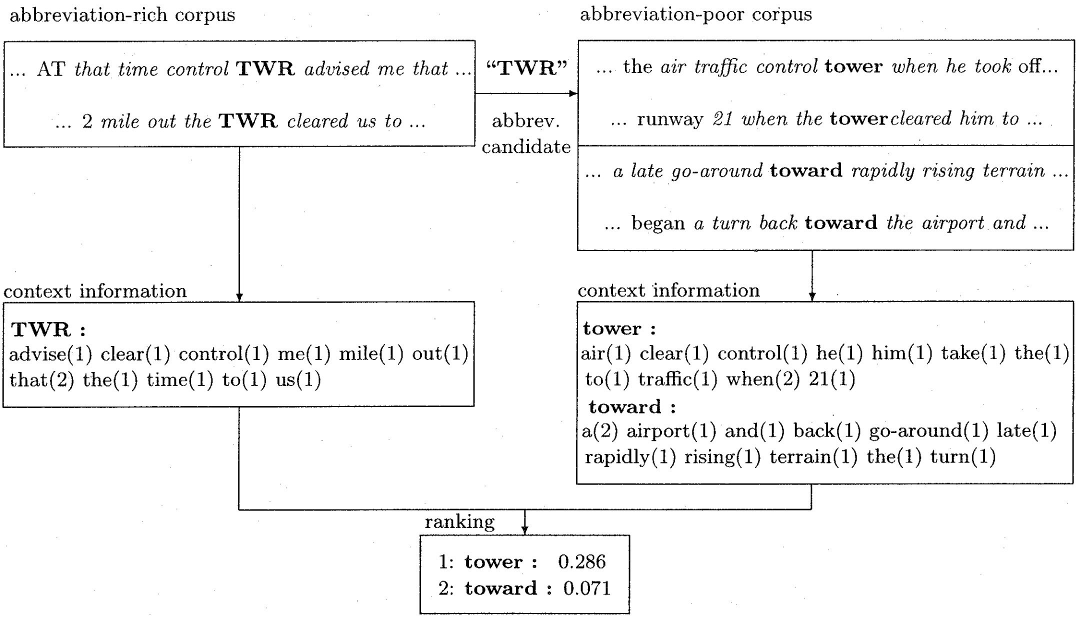
abbreviation-rich corpus
... AT that time control TWR advised me that ...
... 2 mile out the TWR cleared us to ...
"TWR"
abbrev. candidate
abbreviation-poor corpus
... the air traffic control tower when he took off...
... runway 21 when the towercleared him to ...
... a late go-around toward rapidly rising terrain ...
... began a turn back toward the airport and ...
context information
TWR :
advise(1) clear(1) control(1) me(1) mile(1) out(1) that(2) the(1) time(1) to(1) us(1)
context information
tower :
air(1) clear(1) control(1) he(1) him(1) take(1) the(1) to(1) traffic(1) when(2) 21(1)
toward :
a(2) airport(1) and(1) back(1) go-around(1) late(1) rapidly(1) rising(1) terrain(1) the(1) turn(1)
ranking
1: tower : 0.286
2: toward : 0.071

Figure 2: An example of context information

Term weights are assigned to documents and queries using either *tf* (term frequency) or *tf-idf* (term frequency-inverse document frequency). In information retrieval, *tf* contributes to recall and *idf* is used to improve precision. *tf* and *idf* are usually combined by multiplying *tf* by *idf* denoted by *tf-idf*. On weighting the words, we evaluated both *tf* and *tf-idf* in our preliminary test, and found no noticeable difference between them. We thus chose to use *tf* for weighting. [Hea97] also posited that *tf-idf* is not accurate when comparing adjacent pieces of text, and *tf* seems more robust for this purpose. We think that using *idf* may degrade performance because rare words which occur around abbreviations or abbreviation expansion candidates are not representative of them. Also, regarding abbreviations as types and collecting all words around a given type may cancel out rare words.

2.2 Ranking

To rank candidate words, the Vector Space model [Sal88] is used, in which documents and queries are represented in a multi-dimensional space. Each dimension corresponds to a document/query term.

In measuring vector similarity, we used the cosine metric.

$$score = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}},$$

where x_i and y_i are elements appearing around the abbreviation and the abbreviation expansion candidate, respectively.

[Kuk92a] reported that the cosine metric was best in terms of overall effectiveness and efficiency for spelling correction.

2.3 System decision

Abbreviation expansion candidates are ranked by the above score. Then the system has to choose the final answer from amongst these ranked abbreviation expansion candidates, or judge that the original word is not an abbreviation. To do this, we utilize various rules as follows:

1. Rule 1
   The highest scoring abbreviation expansion candidate is the original word form of the abbreviation.
2. Rule 2
   As [UMG93] pointed out English speakers tend to truncate (chop off a word-final substring) or contract (remove internal characters), when abbreviating words. We make use of these features and use the C4.5 [Qui93] decision tree based classification system to derive the effective rules using the 9 features described below.
   (a) First character match:
   Abbreviations usually begin with the same character as the original word form, except when the abbreviation begins with "X", e.g., "XING – crossing".
   (b) Final character match:
   In many cases, the abbreviation and the original word form share the same final character.
   (c) Truncation:
   e.g., "CAPT – captain"

(d) Missing vowels:
The lack of vowels in the abbreviation over the abbreviation expansion candidate provides a strong indication of abbreviation,
e.g., "DSCNT – descent"

(e) Single substring with vowels and consonants:
An abbreviation which is derived from the original word form by removing a single substring, including both vowels and consonants provides a strong indication of abbreviation,
e.g., "CLB – cl<u>im</u>b"

(f) Single substring with consonants:
An abbreviation which is derived from the original word by removing a single substring including only consonants provides a strong indication an inappropriate abbreviation expansion candidate,
e.g., "CLIB – cli<u>m</u>b"

(g) Score:
Score of the cosine metric.

(h) ASRS frequency:
The abbreviation frequency in the ASRS dataset.

(i) NTSB frequency:
The abbreviation expansion candidate frequency in the NTSB dataset.

3. Rule 3
In addition to rule 2, we replace any initial Xs with "TRANS", "CROSS" and "OUT" to find the candidate word.

4. Rule 4
In the aviation domain, acronyms too are used frequently. However acronyms are used consistently in every text, e.g. "GS – Glide Slope". In addition to rule 3, we use a fixed acronym list which contains 114 acronyms, to distinguish between abbreviations and acronyms.

5. Rule 5
In the aviation domain, IATA (International Air Transport Association) Airport 3-letter codes are used commonly. In addition to rule 4, we use IATA airport codes to distinguish between abbreviations and airport codes.

3 Evaluation

We used 10,000 ASRS documents in our experiment (Table 1). In these documents, 260 different types of abbreviations were found by manually check through unknown words detected by the system. However other abbreviations are also included in the ASRS dataset, which were not identified as unknown words by the tagger.

We chose 20 documents, which includes 106 different types of abbreviations, for test set. Among these 106 abbreviation types, 79 abbreviations were detected as abbreviation expansion candidates by the system, 5 abbreviations had no abbreviation expansion candidate in the NTSB dataset and 22 abbreviations were not detected abbreviation expansion candidates, because the tagger did not analyze them as unknown words. The abbreviation "MI" corresponds to both "miles" and "minutes" in the 20 document samples (hypothesis 4 does not hold). Thus we have a total of 107 different abbreviations in the documents. As the system did not detect 22 of these as unknown words, these 22 words are not included in the 260 different types of abbreviations which the system detected. We used the remaining 176 abbreviations for training. In this way, we avoided including the same types of abbreviation in both training and test.

To evaluate the system, we use precision and recall. Precision and recall are defined as below:

$$precision = \frac{w}{w+x+y}$$

$$recall = \frac{w}{w+x+z},$$

where w is the number of the abbreviation types which are correctly disabbreviated; x is the number of the abbreviation types which are incorrectly disabbreviated; y is the number of the abbreviation types which are not actually abbreviations, but incorrectly disabbreviated by the system; and z is number of the abbreviation types which are not detected as abbreviations by the system.

To evaluate our system, we experimented with the various rules described in section 2.3. Using rule 5, recall degrades because the abbreviation inconsistency occurs (hypothesis 3 does not hold), e.g. "SVC" stands for both "service" and an airport name. Talbe 2 shows the result of 20 documents.

Table 2: Results of 20 documents

	precision (%)	recall (%)
Rule 1	39.7 (54/136)	50.5 (54/107)
Rule 2	60.6 (60/99)	56.1 (60/107)
Rule 3	61.6 (61/99)	57.0 (61/107)
Rule 4	67.8 (61/90)	57.0 (61/107)
Rule 5	80.0 (52/65)	48.6 (52/107)

We generated evaluation data for these 20 documents by way of three annotators who are familiar with the aviation domain. Annotators precision ranged from 85.7% to 94.9% and annotators recall ranged from 50.4% to 78.5%, the average annotators precision was 89.4% and the average annotators recall was 64.8%. These result indicate that persons who are familiar with this domain can decipher abbreviations better than our system by 21% in precision and 7% in recall.

To avoid tagger errors, we manually labeled the misrecognized 22 words as "unknown word". The results in Table 3 show that precision is almost identical, but recall is about 10% higher. So we can say that if the tagger is constructed for the specific domain, recall could be improved without degrading precision.

Table 3: Results of 20 documents after relabelling 22 undetected abbreviations words as unknown words

	precision (%)	recall (%)
Rule 1	40.5 (64/158)	59.8 (64/107)
Rule 2	60.3 (73/121)	68.2 (73/107)
Rule 3	61.2 (74/121)	69.2 (74/107)
Rule 4	65.8 (73/111)	68.2 (73/107)
Rule 5	75.3 (58/77)	54.2 (58/107)

To evaluate system performance over the total document set, we define recall* as below, because we cannot check the whole dataset for abbreviations that are not recognized as unknown words by the tagger.

$$recall* = \frac{w}{w+x}$$

As mentioned above, if the tagger were constructed for the given domain, recall* would be equal to recall. Table 4 shows this results.

Table 4: Results for overall document set

	precision (%)	recall* (%)
Rule 1	24.2 (103/426)	39.5 (103/261)
Rule 2	50.0 (110/220)	42.1 (110/261)
Rule 3	52.1 (111/213)	42.5 (111/261)
Rule 4	56.3 (111/197)	42.5 (111/261)
Rule 5	70.6 (96/136)	36.8 (52/261)

Next, we compare our results with previous results. [Too00] performed a two-stage disabbreviation task, i.e. identifier and expander, using the ASRS dataset. [Too00] reported that 188 (different) unknown words were detected at the abbreviation identifier stage. Of the 188 unknown words, some were not abbreviations, but incorrectly predicted by the abbreviation identifier. [Too00] reported that there were only a few errors. Recall of 58.5% is reported in the case that the abbreviation expander returns only one abbreviation expansion candidate. [Too00] does not evaluate precision, but we estimate that precision is few percent lower than recall, because the abbreviation identifier makes a few isolated mistakes. Our system recall is about the same, but precision is 5-10% higher than the results of [Too00] (Table 5).

Table 5: Comparison with previous work (1)

	precision (%)	recall (%)
[Too00]	< 58.5	58.5
Ours (rule 3)	61.6	57.5
Ours (rule 4)	67.8	57.5

[RL95] proposed a system in which the system and user work interactively. That is, the system proposes abbreviation expansion candidates one by one up to five candidates. The user judge whether each candidate is the original word form of the abbreviation in question. [RL95] did a disabbreviation task using photograph captions, and designed their system to accept up to two-word phrases. [Too00] recalculated her results to make them comparable and we also recalculated our results in the same way. Our system recall is about 7-8% lower than the previous results, but precision is between 16-60% higher (Talbe 6).

Table 6: Comparison with previous work (2)

	precision (%)	recall (%)
[RL95]	14.9	71.1
[Too00]	32.8	69.6
Ours (rule 3)	48.6	62.6
Ours (rule 4)	74.4	62.6
with tagger error correction		
Ours (rule 3)	48.0	76.6
Ours (rule 4)	73.0	75.7

If the tagger were tuned to the domain, recall would also be higher than previous results (Table 6 lower part).

4 Conclusion

In this paper, we proposed a disabbreviation method using context information and furnished encouraging result. For example, "S" was successfully disabbreviated to "south" from among 2,725 abbreviation expansion candidates. This could not be achieved without context information, because "S" is only one in character length and has minimalistic character information.

In terms of portability, our method primarily uses context information and we use only general characteristics of English in rule 1 to rule 3, so it may be transferred to other domains easily. Without domain specific knowledge, we can achieve high precision and recall.

We use an abbreviation-poor corpus instead of using a dictionary to search for abbreviation expansion candidates. By using this technique, our system can obtain domain specific information automatically. The abbreviation-poor corpus used in our experiment includes about 7,700 word types, whereas [Too00] used a dictionary of about 35,000 entries and [RL95] one of about 29,000 entries. Our dictionary contains about 22-27% entries of the previous research. This means that our dictionary is tuned to the domain.

Abbreviation inconsistency (the same abbreviation corresponding to different words) may be resolved by using context information from each abbreviation token (not abbreviation type), but we have not experimented with this idea yet. In this case, part-of-speech information for unknown words in the abbreviation-rich corpus and part-of-speech information for abbreviation expansion candidates in the abbreviation-poor corpus may be useful.

References

[FBY92] William B. Frakes and Ricardo Baeza-Yates. *Information Retrieval.* Prentice Hall, 1992.

[Hea97] Marti Hearst. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23:33–64, 1997.

[Kuk92a] Karen Kukich. Spelling correction for the telecommunications network for the deaf. *COMMUNICATIONS OF THE ACM*, 35(5):80–90, 1992.

[Kuk92b] Karen Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, 1992.

[Qui93] J. Ross Quinlan. *C4.5:PROGRAMS FOR MACHINE LEARNING.* Morgan Kaufmann Publishers, 1993.

[RL95] Neil C. Rowe. and Kari Laitinen. Semi-automatic disabbreviation of technical text. *Information Processing & Management*, 31(6):851–857, 1995.

[Sal88] G. Salton. *Automatic Text Processing.* Addison-Wesley, 1988.

[Sch94] Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49, 1994.

[Sch95] Helmut Schmid. Improvements in part-of-speech tagging with an application to German. In *EAL SIGDAT workshop*, 1995.

[Too00] Janine Toole. A hybrid approach to the identification and expansion of abbreviations. In *Proceedings of RIAO'2000*, volume 1, pages 725–736, 2000.

[UMG93] Ramasamy Uthurusamy., Linda G. Means., and Kurt S. Godden. Extracting knowledge from diagnostic databases. *IEEE Expert*, 8(6):27–38, 1993.