Chapter 4

Term Variation

One of the important issues related to terms in a medical QA system is the detection of term variation, which is defined as alternative names for concept, the same as synonymy. This issue arises due to, among others, the different practices in the designation of medical terms among expert and general users. For example, it is common to use Latin and Greek words for medical terminologies among medical experts but this practice is not common among general users, who prefer using popular terms. The differences may lead to the designation of a concept using different terminologies which is known as term variation. As a result, when posing a question to a QA system, a general user may use a term which is different from the one used in the technical corpus. In order for the system to find an answer to such a question, the system should be able to indicate that particular terms in a corpus are variants of a particular term in the question.

In this chapter, we aim at answering research question #4 about the types of term variation in Dutch medical questions and how we can recognize some of the variation types in Dutch documents. We investigate the types of variation which commonly occurs in medical questions in Section 4.2. Before we describe our experiments, we explain the document pre-processing in Section 4.3 and the evaluation methodology in Section 4.4. In Section 4.5 and 4.6, we in turn describe our methods and experiments to extract synonyms and abbreviations. We summarize this chapter in Section 4.7.

4.1 Introduction

Term variation can be defined as alternative names for a concept. For example, carcinoma and cancer are variants (of synonym type) and can be used interchangeably. Depending on the domain, terminological variation in text is estimated around 15-35% of their occurrence (Daille, 2003). It is a well-known phenomenon that needs special treatment.

There are various applications where recognition of term variation is important, such as indexing and retrieval (Jacquemin, 2001), conceptual structuring (Daille, 2003), and enhancing an ATR method (Nenadić et al., 2004). Term variation also occurs in controlled terminological resources, such as UMLS.

Consider the previous C-value method that originally treats term variants,
e.g. blood cell and cell of blood, as different candidate terms. This keeps the frequency of each term unchanged. As a consequence, the C-value scores of the terms are lower compared to a situation where the joined frequency of the variants is used to calculate the scores. To enhance the scores, Nenadić et al. (2004) recognize and normalize term variants, and then use the joined frequency of the variants to compute the C-value of each term variant. Their experiment on a biomedical corpus shows an improvement of precision by 20–70%.

Nenadić et al. (2004) define several types of term variation, namely orthographic, morphological, lexical, structural, and acronyms and abbreviation variation.

Orthographic Variation This variation can be detected from the usage of specific writing rules or symbols in the surface representation of terms. The rules or symbols and their examples of term variants taken from the UMLS are given in Table 4.1.

<table>
<thead>
<tr>
<th>Rules</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>hyphens and slashes</td>
<td>8-Aminonaphthalene, 8 Aminonaphthalene</td>
</tr>
<tr>
<td>lower and upper cases</td>
<td>para-Aminobenzoic Acid, Para-aminobenzoic acid</td>
</tr>
<tr>
<td>spelling variation</td>
<td>Accident-prone behavior, Accident-prone behaviour</td>
</tr>
<tr>
<td>different Latin/Greek</td>
<td>Antiestrogen, Antioestrogen</td>
</tr>
<tr>
<td>transcriptions</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1: Some examples of orthographic variation.

Normalizing these variants can be done through simple heuristics, for example, by replacing hyphens with spaces or by replacing specific characters of neoclassical forms with a character (e.g., oe → e).

Morphological Variation Word formation can lead to variation, for example, in the inflectional phenomena (e.g., a plural form of Antiestrogen → Antiestrogens) and derivational transformation (inactivated vaccine and deactivated vaccine). The normalization of the inflection is based on PoS information by changing all forms into their singular and non-possessive forms.

Lexical Variation Synonymous lexical items that may be used interchangeably are variants, such as veteranenziekte and legionairsziekte ‘legionnaires’ disease’. Thesaurus or dictionaries containing synonyms are the main sources for normalizing this kind of variation.

Structural Variation This variation type can be constructed from the possessive usage of nouns using prepositions (e.g., tumor of abdomen and abdominal tumor), prepositional variants (e.g., bleeding from mouth and bleeding in mouth), and term coordinations (Acquired chest and rib deformity). The prepositional terms can be normalized into expressions without prepositions. For example, genes of human is normalized into human genes. Term coordinations are more difficult to resolve, even for humans, due to their ambiguity.
4.1. Introduction

Acronyms and Abbreviation Variation

In biomedicine, this variation is very frequent. For example, AIDS is an abbreviation of the term Acquired Immune Deficiency Syndrome. The normalization of this kind of variation is mainly based on orthographic and syntactic information of their contexts. A more detailed review on acronym recognition has been reported in Krauthammer and Nenadic (2004).

The handling of term variation is mainly based on rules and heuristics that usually can be created by investigating the linguistic typology of candidate terms in text. It is motivated by practical purposes rather than linguistic consideration (Maynard, 2000).

In our task, the purpose of the variation recognition is mainly to detect synonyms that frequently occur in our extracted relations. These synonyms are linked by coordinations that need to be separated correctly. Consider, for example, the following sentence that contains a medical relation in our corpus:

(1) Lepra is een besmettelijke ziekte die wordt veroorzaakt door de leprabacil “Mycobacterium leprae” of “bacil van Hansen” genoemd naar de Noorse arts Armauer Hansen die deze in 1873 ontdekte.

‘Leprosy is an infectious disease caused by the leprabacil “Mycobacterium leprae” or “Hansen’s bacillus” named after the Norwegian physician Armauer Hansen who discovered it in 1873.’

The sentence contains variants of the term leprabacil, namely Mycobacterium leprae and bacil van Hansen. Without extraction of individual terms, the extracted relation will link a single term to multi terms, such as:

(2) [Leprosy] [is caused by] [Mycobacterium leprae or Hansen’s bacillus]

This relation will be able to answer the following question:

(3) What is the cause of leprosy?

but not these questions:

(4) a. What disease is caused by Hansen’s bacillus?
   b. What is another name for Hansen’s bacillus?

Since the term Hansen’s bacillus is not a complete argument of the relation in the example (2), but only a partial one, finding an answer based on a table look up strategy will face a difficulty, unless it applies partial matches. This problem could be be solved by recognizing structural variation, especially via the extraction of individual terms from coordinations.

In medical text, there is another frequent variation type, i.e the acronyms and abbreviation variation. Consider, for example, the following sentence from our corpus:

(5) Tuberculose, afgekort met TBC, of zelfs TB, is een vooral vroeger zeer gevreesde infectieziekte die wordt veroorzaakt door de bacterie “Mycobacterium tuberculosis”.

‘Tuberculosis, abbreviated with TBC, or even TB, is formerly a very dreaded disease caused by the bacterium “Mycobacterium tuberculosis”.’
This sentence obviously shows that acronyms lead to term variations. In a QA task, a user may use any variant he/she likes in questions. Therefore, our term variation recognition system in Chapter 4 is also aimed at recognizing this type of variation.

4.2 Types of Variation in Medical Questions

Before we detect variation for a medical QA system, first we investigate the types of term variation frequently occurring in users' questions. Then, we apply several methods to identify the variation. The detected variation can be used to map a term used in a question to its variation used in a corpus.

For the purposes of the investigation, we collected a set of medical questions from the Internet using a Web service provided by a major search engine. We created two phrasal queries, namely:

(6) a. “wat is”
   ‘what is’
b. “wat zijn”
   ‘what are’

which were aimed at getting search results containing medical questions starting with those strings, such as, Wat is de oorzaak van ...? ‘What is the cause of ...?’ and Wat zijn de symptomen van ...? ‘What are the symptoms of ...?’ To avoid non-medical questions from the results, we limited the source domains for the queries to only 27 medical Web sites in Dutch, such as forum.dokter.nl, www.ggdgroningen.nl, and www.gezondheidsplein.nl. Most of the Web sites are targeted to general users and on some Web sites the users can post their medical questions to experts.

We submitted two queries to each Web site and retrieved a maximum of 100 results for each query. This process resulted in 1800 unique questions which occur from 1 to 98 times in the results. For our investigation, we only used questions occurring at least twice, which resulted in 370 questions. Then, we extracted all terms in the questions by removing all function and stop words, which resulted in 359 unique terms consisting of 221 single-word terms and 138 multi-word terms.

We manually grouped the single-word terms into Dutch terms and non-Dutch terms. We found that around 70% of the single-word terms are in Dutch and the rest are in English, Latin, or Greek. This may lead to synonym variation between technical terms and their popular terms, or between English, Latin, and Greek terms and their translations in Dutch. For example, in the questions in (7), the term diabetes (Latin) has synonyms suikerziekte (Dutch) and diabetes mellitus (Latin).

(7) a. Wat zijn de symptomen van diabetes? (www.kindengezin.be)
   ‘What are the symptoms of diabetes?’
b. Wat is diabetes mellitus? (forum.dokter.nl)
   ‘What is diabetes mellitus?’
c. Wat is suikerziekte? (www.pfizer.be)
   ‘What is diabetes (lit. “sugar disease”)?’

4.3. Pre-processing: Term Tagging

Among the terms in the questions, there are 20 terms having characteristics of abbreviations or containing abbreviations, such as ADHD (Attention Deficit Hyperactivity Disorder), SIS (Shaken Infant Syndrome), TBC (tuberculosis), and XTC gebruik (the use of Ecstasy). As shown in the questions in (8), besides using abbreviations, some questions also use the long forms of abbreviations such as humaan papillomavirus (abbreviated as HPV).

(8)  
a. *Wat is tuberculose?* (gezondheid.be)  
‘What is tuberculosis?’
b. *Wat is TBC?* (ggdgroningen.nl)  
‘What is TBC?’
c. *Wat is een humaan papillomavirus?* (mens-en-gezondheid.infonu.nl)  
‘What is a human papilloma virus?’

The last type of variation is generated by the construction of multi-word terms using coordination, e.g., lactose of melksuiker and effect van Strumazol en Radioactiefjodium, or permutation, e.g., Marfan syndroom and Syndroom van Marfan as shown in the questions in (9). We found 34 multi-word terms that use prepositions (e.g., van ‘of’, bij ‘at’) and coordinations (e.g., of ‘or’, en ‘and’). These two constructions are the most frequent constructions among grammar-based variation in the questions.

(9)  
a. *Wat veroorzaakt het Marfan syndroom?* (www.marfan.nl)  
‘What causes the Marfan syndrome?’
b. *Wat is het Syndroom van Marfan?* (mens-en-gezondheid.infonu.nl)  
‘What is the Syndrome of Marfan?’

In the above investigation we find three variation types which frequently occur in the questions. In the present chapter, we limit our variation extraction to the first two variation types, namely synonym and abbreviation variation. We do not extract the last type of variation since it requires a comprehensive set of grammatical rules to deal with all grammar-based variation such as has been developed in the FASTR system for English (187 rules) and French (110 rules) (Jacquemin, 2001). For Dutch, we leave the recognition of this variation type for the future work.

4.3 Pre-processing: Term Tagging

Before we extract variation from input text, first we have to identify and annotate terms occurring in the text. In our implementation, this task is carried out by the term extraction (ATR) phase from the previous chapter (Chapter 3). Consider, for example, the following input sentence,

(10) *Het geeft tevens pijnstilling (analgesie) en een minimale verslapping (spierrelaxatie).* (Wikipedia)  
‘It also provides pain killing (analgesia) and a minimal relaxation (muscle relaxation).’
Chapter 4. Term Variation

We annotate this sentence with part-of-speech tags using Alpino (van Noord, 2006), identify candidate terms in the sentence using the linguistic filter in Figure 3.10, and then annotate the terms as shown in the following output,

(11) Het/det geeft/verb tevens/adv <pijinstilling/noun> /(punct <analgesie/noun> )/punct en/vg een/det <minimale/adj verslapping/noun> /(punct <spierrelaxatie/noun> )/punct ./punct

In the above example, the sentence is tagged with PoS tags and the identified terms, i.e., *pijinstilling*, *analgesie*, *minimale verslapping*, and *spierrelaxatie*, are marked.

The linguistic filter used is suitable for identifying terms, but not for identifying long forms of some abbreviations as shown in the following example,

(12) Nederlandse/name Vereniging/name voor/name Addison/name en/vg Cushing/name Patienten/name /(punct NVACP/name )/punct de/det vereniging/noun waarbinnen/pp de/det Werkgroep/name Conn/name actief/adj is/verb ./punct

‘Dutch Association for Addison and Cushing Patients (NVACP), the association where the Working Group Conn is active.’

The above PoS-tagged sentence contains an abbreviation, i.e. *NVACP*, and its long form, i.e. *Nederlandse Vereniging voor Addison en Cushing Patiënten* ‘Dutch Association for Addison and Cushing Patients’. Since this long form contains the coordination *en* ‘and’ which is tagged as *vg*, the linguistic filter will only be able to capture its substrings, i.e. *Nederlandse Vereniging voor Addison* and *Cushing Patiënten*, but not the whole string.

To solve this problem, we modify the linguistic filter by allowing coordinations (vg), which frequently occur in long forms, to appear in the variation. The modified linguistic filter is shown in the following figure.

```
(((^\)+/(adj|name|noun) )+)(([^ ]+\/(adj|name|noun) )+)\n([^ ]+\/(name|noun) )\n([^ ]+\/(prep|vg) )\n([^ ]+\/(det )\n([^ ]+\/(adj|name|noun) )\n([^ ]+\/(name|noun))
```

Figure 4.1: The PoS-tag filter for detecting the abbreviations’ long forms. This is a one line regular expression filter.

The above filter is only applied for detecting abbreviation and long forms that will later be filtered and refined.

4.4 Evaluation Methodology

We evaluate the performance of the extraction methods using precision, recall, and harmonic mean measures. Since we do not have a list of all synonym pairs and abbreviation pairs in the corpus, we cannot compute precision and recall based on the set of all true positive pairs. Instead, we compute the measures based on a gold standard, which is a subset of pairs selected randomly from
4.5 Extracting Synonyms

4.5.1 Previous Work

Studies on identifying term variation have mainly focused on terms that are lexically and syntactically related, such as *tumor derived cell* and *cells from tumors* (Jacquemin, 1996; Ibekwe-Sanjuan, 1998), and share many words, such as *cerebrospinal fluid* and *cerebrospinal fluid protein assay* (Hole and Srinivasan, 2000). Their identification methods are restricted to actual strings in the terms and ignores surrounding context words.

As for identifying synonyms, context words play important roles. A well known method of using surrounding contexts to detect lexical relations has been reported in Hearst (1992). The contexts are encoded into lexico-syntactic patterns aimed at extracting hyponymy relations, as illustrated in the following example:

(13)  *such NP as {NP ,} *(or|and))} NP
    .. works by such authors as Herrick, Goldsmith, and Shakespeare.
    \[\Rightarrow \text{hyponym(“author”, “Herrick”), hyponym(“author”, “Goldsmith”), hyponym(“author”, “Shakespeare”).}\]
Chapter 4. Term Variation

The first line of the above example is one of the lexico-syntactic patterns, the next line is a fragment of an input sentence, and then some predicates that can be derived from the sentence.

The context and pattern can be discovered automatically using a partially-supervised approach, which is aimed at avoiding substantial manual labor for preparing a training data set as commonly required in supervised machine learning techniques. In her approach, Hearst starts with one pair of terms at a time to get new patterns and new hyponymy instances, and then applies a bootstrapping technique to get more patterns and more instances.

Such a partially-supervised approach combined with a bootstrapping technique seems to be an attractive approach. Another well known method that uses the same approach is called DIPRE (Dual Iterative Pattern Relation Extraction), which is aimed at extracting structured relations from the Web pages (Brin, 1999). DIPRE automatically learns binary relation patterns from a seed list of target relations, and then uses the patterns to extract many relation tuples.

Agichtein and Gravano (2000) develop a tool called Snowball, which addresses the same issue (extracting information relationships from text) and even using the same idea as DIPRE. The main contribution of this tool is in using a novel technique for evaluating the quality of the patterns and the tuples generated in each iteration process. Only reliable patterns and tuples are used for the following iterations. At the pattern generation stage, the key difference between Snowball and DIPRE is that Snowball uses named entity tags (e.g., LOCATION, ORGANIZATION, PERSON) in a pattern.

To evaluate the generated patterns and tuples, the Snowball method improves the DIPRE method by computing the confidence score of a pattern. This score is equal to the ratio of the number of positive-matching tuples and the number of all tuples generated by the pattern. A tuple, for example <Intel, Santa Clara> (an organization-location tuple), is considered to have a positive matching if in the previous iteration the organization Intel has a location Santa Clara; otherwise, the tuple will have a negative matching.

The above case is suitable for organization-location relations because an organization usually has only one location as its headquarters. In the case of synonym variation, a term may have more than one variant which makes the above matching method less useful. To adapt Snowball for detecting synonymous terms, Yu and Agichtein (2003) use user-provided examples of synonymous term pairs, and combine the system with a machine learning technique. However, since we only provide a small set of synonym pairs for the initial iteration, and no more positive examples to evaluate the generated tuples, the later method is less suitable for our purpose.

In the same project where we were involved, van der Plas (2008) compared three different distributional similarity methods, namely syntax-based, alignment-based, and proximity-based methods, to extract lexico-semantic information including synonyms from text. In the syntax-based method, she uses several syntactic relations, i.e., subject, object, adjective, coordination, apposition, and prepositional complement, between two words being investigated for their similarity. In the alignment-based method, she uses parallel corpora to get translation pairs that are found in two directions. While in the proximity-based method, she defines the context word of a headword in a sentence with respect to a window of a particular number of words around the headword. She
compared the performance of the various combinations of weight functions, i.e., frequency, MI, and t-test, compared the similarity measures between word vectors (Dice and Cosine). She found that the alignment-based method is better at finding synonyms than the syntax-based method. When limited text data is available, proximity-based method would be a better choice. However, she only considered single-word units in her experiments, while in this thesis we would like to extract synonyms of single-word and multi-word units.

Another method aimed at evaluating synonyms from distributionally similar words has been reported in Lin et al. (2003). They use a patterns-of-incompatibility method to evaluate the synonym score of a \( x-y \) pair by comparing the number of hits where \( x \) and \( y \) occur near each other in the Web pages against the number of hits where \( x \) and \( y \) occur following a set of incompatibility phrasal patterns in the Web pages. The numbers are retrieved from the search engine AltaVista\(^2\). To get the number of the compatibility hits of a pair, the following query is submitted to the search engine:

\[
(14) \quad x \text{ NEAR } y
\]

and to get the number of incompatibility hits, the following phrasal patterns are used:

\[
(15) \quad \begin{array}{l}
\text{a. "from } x \text{ to } y\text{"} \\
\text{b. "from } y \text{ to } x\text{"} \\
\text{c. "either } x \text{ or } y\text{"} \\
\text{d. "either } y \text{ or } x\text{"}
\end{array}
\]

Having the numbers of hits, the compatibility score of a \( x-y \) pair is defined as:

\[
\text{score}(x, y) = \frac{\text{hits}(x \text{ NEAR } y)}{\sum_{\text{pat} \in P} \text{hits(}\text{pat}(x, y)\text{)) + \epsilon}} \tag{4.4}
\]

where \( \text{hits(query)} \) is the number of hits returned by the search engine for the query, \( P \) is the set of query patterns in (15), \( \text{pat}(x,.) \) is a regular expression pattern using \( x \) and \( y \), and \( \epsilon \) is a constant to avoid a zero denominator. A \( x-y \) pair is classified as a synonym pair if its score is higher than a threshold \( (\theta=2000) \). Compared to using bilingual dictionaries for identifying synonyms among word pairs, this method shows a significantly higher recall (95.0% v.s. 39.3%) although experiencing a lower precision (86.4% v.s. 93.9%).

The problem faced by the above method which selects synonym pairs among a set of term pairs extracted from a corpus is similar to ours. The adopted solution is straightforward regardless the size of the corpus, because it uses web data. Considering that the size of our corpus is relatively small, we decided to use this approach to evaluate extracted synonym pairs from our corpus.

### 4.5.2 Extracting Candidate Synonyms

Our method to extract synonyms is adapted from the DIPRE method (Brin, 1999) and we incorporate syntactic information into its patterns, as in Hearst (1992) and Pustejovsky et al. (2001). The syntactic information (PoS tags) is mainly for detecting terms occurring in the corpus.

---

\(^2\)AltaVista [http://www.altavista.com/]
Figure 4.2 shows the diagram of the synonym extraction method which consists of three consecutive processes within an iteration. The extraction is started with the injection of a small set of synonym pairs as seeds which could be selected manually from the corpus. Each pair is a tuple of the following format,

\[(\text{term}1, \text{term}2)\]

where \text{term}1 is the first argument and \text{term}2 is the second argument of a synonym pair. Having a seed list, we apply the following steps to learn variation patterns, and finally to extract synonym pairs from the corpus:

\textbf{Step 1} The \textit{Pattern Learning} process searches for the occurrence of the seed tuples in the corpus and keeps the contexts surrounding the tuples. The contexts of a tuple consist of a prefix (a word string before the first argument), a middle (any word strings between the arguments), and a suffix (a word string after the second argument). Then it generates the following pattern,

\[
<\text{prefix}, [\text{arg}1], \text{middle}, [\text{arg}2], \text{suffix}>
\]

\textbf{Step 2} The \textit{Tuple Extraction} process applies the generated patterns over the corpus to extract a new set of candidate synonym tuples that match with the patterns.

\textbf{Step 3} The \textit{Tuple Evaluation} component evaluates the tuples by computing their semantic compatibility based on their occurrence in the Web pages. The detailed method for this evaluation is provided in Section 4.5.3. The output of this evaluation is ranked based on their compatibility scores. Tuples with scores above a threshold are considered as synonym pairs.

\textbf{Step 4} The last step uses the extracted synonym pairs as a new seed list for
4.5. Extracting Synonyms

the next iteration. Repeat Step 1 to Step 3 until no new synonym pairs are extracted.

As in Hearst (1992) and Brin (1999), the size of the initial seed list can be very small, such as 3 pairs. However, if the size of the corpus is small like ours, the occurrence of the seeds in the corpus can be very low as well. They will generate a small number of synonym patterns, and subsequently the patterns will extract a small set of new synonym tuples. Two possible solutions for this problem are to increase the size of the initial seed list or to increase the number of iterations until there are no more new tuples added into the set of the extracted synonym pairs. The second solution is not plausible if the size of the corpus is large.

Starting from the second iteration, we can also evaluate the generated patterns in Step 1 using the compatibility scores of their supporting synonym pairs that have been computed in Step 3 of its preceding iteration. However, in this experiment, we do not apply such evaluation, since the size of the corpus is small and the extracted synonym pairs has been evaluated in Step 3.

4.5.3 Evaluating Synonyms

The output of Step 3 in the above method is a set of candidate synonym pairs. To identify true synonym pairs, we follow a method described in Lin et al. (2003), which use Web pages as an external source to measure the synonym compatibility hits of each pair. We use the following phrasal patterns \( P_{or} \) to get the number of synonym compatibility hits:

\[
\begin{align*}
(17) & \quad \text{a. } \text{"}x \text{ of } y\text{"} \\
& \quad \text{\text{'}x or } y\text{\'} \\
& \quad \text{b. } \text{"}y \text{ of } x\text{"} \\
& \quad \text{\text{'}y or } x\text{\'} \\
& \quad \text{c. } \text{"}x (y)\text{"} \\
& \quad \text{d. } \text{"}y (x)\text{"}
\end{align*}
\]

and the following phrasal patterns \( P_{and} \) to get the number of synonym incompatibility hits:

\[
\begin{align*}
(18) & \quad \text{a. } \text{"}x \text{ en } y\text{"} \\
& \quad \text{\text{'}x and } y\text{\'} \\
& \quad \text{b. } \text{"}y \text{ en } x\text{"} \\
& \quad \text{\text{'}y and } x\text{\'}
\end{align*}
\]

The intuition behind the patterns is that if two terms are synonyms to each other, they may cooccur in the Web pages linked by an \textit{of} ‘or’ coordination like in the sentence \textit{Suikerziekte of diabetes is een chronische aandoening ‘Sugar disease or diabetes is a chronic disease’}, or by parentheses such as \textit{abnormale verbinding (fistel) ‘abnormal connection (fistula)’}. Unfortunately, parentheses will be ignored by search engines making the use of parentheses in query terms meaningless for the search engine. We will explain how to solve this problem shortly.

On the other hand, if two terms are not synonymous, they could be linked by the \textit{en} ‘and’ coordination like in the sentence \textit{Wat zijn de verschillen tussen...}
**Query** | **Hits**
---|---
"suikerziekte of diabetes" | 2030
"diabetes of suikerziekte" | 1580
"suikerziekte (diabetes)" | 4200
"diabetes (suikerziekte)" | 41745
"suikerziekte en diabetes" | 8
"diabetes en suikerziekte" | 3060
"bacteriën of virussen" | 1410
"virussen of bacteriën" | 1010
"bacteriën (virussen)" | 0
"virussen (bacteriën)" | 0
"bacteriën en virussen" | 10500
"virussen en bacteriën" | 9070

Table 4.3: Examples of phrase queries and their numbers of hits for two synonym pairs.

*bacteriën en virussen?* ‘What are the differences between *bacteria* and *viruses*?’ Since the semantic relation of a synonym pair is bidirectional, the evaluation should be carried out in two directions by exchanging the positions of the terms in the queries.

For each synonym pair, four queries using the patterns in (17) and two queries using the patterns in (18) are submitted to the Yahoo! Search Web Services. For each query we only keep the total number of hits returned by the search engine. And particularly for the patterns in (17-c) and (17-d) which use parentheses, we also retrieve the snippets of the first 100 results and then count the number of snippets that match with the following regular expression:

```
/(x\s+(\s*y\s*(\)|,\s+))\s+)/i
```

meaning that *x* and *y*, the first and the second terms in a synonym pair, should be separated by an opening parenthesis and closed by a closing parenthesis or a comma. We then use the following formula to get an approximated number of hits for a query pattern:

$$
\text{hits}(x,y) = \begin{cases} 
  f(regex(x,y)) & \text{if } f \leq \text{hits} \\
  \frac{f(regex(x,y))}{100} \times \text{hits} & \text{otherwise}
\end{cases}
$$

where \(f(regex(x,y))\) is the number of snippets matching the above regular expression for a *x-y* term pair and \(\text{hits}\) is the total number of hits returned by the search engine.

As an illustration, consider Table 4.3 that shows the submitted queries for the term pairs *<suikerziekte, diabetes>* and *<bacteriën, virussen>* and the numbers of hits returned by the search engine. For the first term pair, its first four phrase queries which use patterns in (17) have a total number of hits which is higher than the total number of hits of the other phrase queries using patterns in (18). This shows that *suikerziekte* and *diabetes* pair is a likely synonym pair.

Is the term *bacteriën* ‘bacteria’ a synonym of the term *virussen* ‘viruses’? We can reveal the answer from information in the second row of the table. The
total number of hits for the last two phrase queries which use patterns in (18)
is higher than the total number of hits for the other phrase queries which use
patterns in (17). In the other words, the terms bacteriën and virussen are more
frequently linked by en ‘and’ coordinations. This indicates that this term pair
probably is not a synonym pair.

Given the number of hits for each phrase query, we define the compatibility
score of a synonym pair as follows:

\[ \text{score}(x, y) = \frac{\sum_{\text{pat} \in P_{or}} \text{hits}((x, y))}{\sum_{\text{pat} \in P_{and}} \text{hits}((x, y)) + \epsilon} \]  
(4.6)

where \( P_{or} \) and \( P_{and} \) are phrasal patterns in (17) and (18), respectively, \( \text{hits(query)} \)
is the number of hits returned by the search engine for the query, \( \text{pat}(x, y) \) is
a query for the term \( x \) and \( y \) using the pattern \( \text{pat} \), and \( \epsilon \) is a small constant
to prevent the zero value of the formula’s denominator; we set \( \epsilon=0.1 \). A term
pair is classified as a synonym pair if the value of its score is above a threshold
\( \theta \); we set \( \theta=30 \). Moreover, since the semantic relation of a synonym is bidirec-
tional, we also require the total numbers of hits of each direction in patterns
(17) should be higher than 0.

Besides using the above phrasal patterns, we also use the following ‘NEAR’
phrasal patterns \( (P_{\text{NEAR}}) \) from Lin et al. (2003) to get the number of synonym
compatibility hits:

\[ x \text{ NEAR } y \]  
(19)

We will investigate which synonym compatibility patterns, i.e. (17) or (19),
show the best performance.

4.5.4 Experiments and Results

We evaluate the synonym extraction method on the IMIX medical corpus which
contains 57,004 sentences. These sentences have been parsed using the Alpino
parser, and have been annotated with candidate terms identified using the lin-
guistic filter in Section 4.3.

As a seed list, we randomly select three synonym pairs from the corpus,
namely:

<netvlies, retina>
<tussenschot, septum>
<poliep, vleesboom>

In Step 1, we search for the occurrence of these seeds in the corpus, and then
generate their patterns. As an illustration, the followings are patterns and their
frequency generated by the above seeds:

3 het/det [t] (/punct [t] )/punct
1 het/det [t] of/vg [t] is/verb
1 het/det [t] (/punct het/det [t] )/punct
1 [t] (/punct [t] )/punct
1 (/punct [t] )/punct tussen/prep de/det boezems/noun ,/punct \ 
het/det septum/noun tussen/prep de/det kamers/noun en/vg \ 
het/det [t] tussen/prep
Table 4.4: Numbers of synonym pairs extracted at each iteration of the two experiment settings.

<table>
<thead>
<tr>
<th>Iteration</th>
<th>( P_{or} )</th>
<th>( P_{NEAR} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>35</td>
<td>63</td>
</tr>
<tr>
<td>2</td>
<td>53</td>
<td>98</td>
</tr>
<tr>
<td>3</td>
<td>137</td>
<td>299</td>
</tr>
<tr>
<td>4</td>
<td>206</td>
<td>499</td>
</tr>
<tr>
<td>5</td>
<td>229</td>
<td>636</td>
</tr>
<tr>
<td>6</td>
<td>233</td>
<td>654</td>
</tr>
<tr>
<td>7</td>
<td>233</td>
<td>699</td>
</tr>
<tr>
<td>8</td>
<td>233</td>
<td>699</td>
</tr>
</tbody>
</table>

In Step 2, we use the patterns to extract new synonym pairs from the corpus, and then in Step 3 we evaluate the synonym pairs using two settings of phrase queries, namely \( P_{or} \) and \( P_{NEAR} \). After calculating the compatibility score of each synonym pair in Step 3, in the last step we select a subset of synonym pairs from each of the setting output whose scores are above the following thresholds: 30 for the \( P_{or} \) setting and 100 for the \( P_{NEAR} \) setting. The final output of the first iteration are two seed lists, which contain 35 synonym pairs for the \( P_{or} \) setting and 63 synonym pairs for the \( P_{NEAR} \) setting.

To get a high coverage of synonym pairs, we run the synonym extraction cycles up to 8 iterations. Table 4.4 shows the numbers of term pairs labelled as synonyms at each iteration for each setting. From the same initial seed list, the \( P_{NEAR} \) setting returns more pairs than the \( P_{or} \) setting. The highest coverage of the first and the second settings are achieved at the 6th and 7th iterations, respectively. If the number of the initial seeds is high, which is not difficult to provide, the highest coverage can be achieved in fewer iterations.

Table 4.5 shows 40 examples of synonym pairs from three blocks of ranks extracted at the last iteration of each setting. The first 10 ranks of the results are filled with the same synonym pairs, where all of them have large total numbers of hits for synonym compatibility patterns in (17) and (19) and zero hit for synonym incompatibility patterns in (18). This shows that the Web is a valuable resource for evaluating the compatibility of synonym pairs, and that comparing the numbers of hits of the opposite queries is a good measure for scoring the strength of the pairs.

Most of the synonym pairs in the first block (or the first 20 ranks) of both settings are of abbreviation pairs. These results indicate that the query patterns and the scoring formula can also be used to evaluate abbreviation pairs. Compared to synonym pairs, abbreviation pairs occur more frequently in those query patterns. Based on this finding, we will evaluate the application of this method for the extraction of abbreviation pairs in the next experiments.

Most of the errors, such as \( <vagina, vaginitis> \), \( <hart, endocarditis> \), and \( <rug, ruggenpriek> \), are caused by imprecision in extracting multi-word terms that contain determiners \( de \) or \( het \). Since the linguistic filter used for extracting terms does not consider a determiner as a part of a multi-word term, the filter may extract an incomplete substring of a term. For example, the term \( vagina \) is an incomplete substring of the term \( onsteking van de vagina \) 'inflammation of the vagina'.
Table 4.5: Examples of synonym pairs extracted using the $P_{or}$ and $P_{NEAR}$ query patterns. The asterisk (*) signs indicate incorrect pairs.
Chapter 4. Term Variation

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$P_{or}$</th>
<th>$P_{near}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>1</td>
<td>1.00</td>
<td>0.15</td>
</tr>
<tr>
<td>2</td>
<td>1.00</td>
<td>0.18</td>
</tr>
<tr>
<td>3</td>
<td>1.00</td>
<td>0.42</td>
</tr>
<tr>
<td>4</td>
<td>0.98</td>
<td>0.68</td>
</tr>
<tr>
<td>5</td>
<td>0.98</td>
<td>0.70</td>
</tr>
<tr>
<td>6</td>
<td>0.98</td>
<td>0.70</td>
</tr>
<tr>
<td>7</td>
<td>0.98</td>
<td>0.70</td>
</tr>
<tr>
<td>8</td>
<td>0.98</td>
<td>0.70</td>
</tr>
</tbody>
</table>

Table 4.6: The performance of the synonym extraction method at each iteration on the two experiment settings.

However, synonym pairs with such incomplete terms can be discarded by evaluating their occurrence in the Web pages using patterns in (17-c) and (17-d). Consider, for example, the term pair $\langle$urine, hematurie$,\rangle$, whose correct term of the first argument is bloed in de urine ‘blood in the urine’. A query which uses pattern in (17-c) gets 480 results containing the string ‘urine (hematurie)’, but using pattern (17-d) the query gets 0 results containing the string ‘hematurie (urine)’. Therefore, this pair will be discarded in the $P_{or}$ setting, but not in the $P_{near}$ setting since queries using pattern (19) get 2890 results.

Regarding the previously mentioned errors, the $P_{or}$ setting failed to discard the pair $\langle$vagina, vaginitis$,\rangle$ because both queries that use patterns in (17-c) and (17-d) return 1304 and 6 results, respectively, and no threshold was applied on the numbers of hits. Looking at the results which have a high discrepancy, we can improve precision by applying a threshold against the number of hits required by each pattern. However, this will reduce recall.

Another error shown in the table is the pair $\langle$Rudolf Steiner, 1861-1925$,\rangle$. This pair contains a relation between a person and a date. Although a number of such pairs were popping up during the extraction of candidate synonyms, most of them can be filtered out by the patterns in (17).

4.5.5 Evaluation

We assume that the last iteration of the synonym extraction cycles produces the most complete list of synonym pairs that can be found in the corpus. Based on this assumption, we generate a list of expected synonym pairs ($E$) from the combined output of $P_{or}$ and $P_{near}$ settings. This list consists of 60 expected true positive pairs ($E+$) and 60 expected true negative pairs ($E-$), which are selected randomly and evaluated manually.

From each iteration, we get a set of synonym pairs extracted by the extraction method. To get a set of observed synonym pairs ($O$), we count the number the synonym pairs which overlap with $E+$ (to get the $O+$ value) and with $E-$ (to get the $O-$ value). Then, we compute the precision, recall, and F-measure values of the extracted synonym pairs.

The performance of the synonym extraction method is presented in Table 4.6. For each setting and iteration, the table provides the measurement values.
4.6 Extracting Abbreviations

The table shows that the $P_{or}$ setting significantly outperforms $P_{NEAR}$ setting in precision at all iterations, while the $P_{NEAR}$ setting outperforms $P_{or}$ setting in recall at most of the iterations. And in the combined performance (F-measures), the first setting outperforms at all of the iterations with the highest value of 0.82 at the 5th iteration.

These results show that the phrase query pattern in (19), which was used in Lin et al. (2003) to evaluate the compatibility of synonymous words, is not suitable for evaluating the compatibility of synonymous terms. This is because two terms cooccurring near each other often do not convey a similar meaning (synonym) but they may construct another relation type, such as a cause-effect relation or a location-disease relation. For example, the term pair ‘ear canal’ and ‘otitis externa’, which were extracted in the $P_{NEAR}$ setting, cooccur 785 times near each other as shown in the following sentence:

(20) Zaemmersoor of otitis externa: is een infectie van de gehoorgang en niet van het middenoor als dusdanig.

‘Swimmers ears’ or otitis externa: is an infection of the ear canal and not of the middle ear as such.’

It is obvious from the sentence that those terms are not related as a synonym pair but as a disease-location pair. Therefore, using the $P_{NEAR}$ patterns and resulting number of pairs of nearby occurrence will lead to misclassifications. A stricter pattern, i.e. “$x$ or $y$”, will only count their frequency if they cooccur in a particular context. Since this pattern, which produces the phrase query “‘ear canal’ and ‘otitis externa’, has no result, thus, the pair $<otitis externa, gehoorgang>$ will not be considered as a synonym pair.

The only pair in the test set that was incorrectly classified by the setting $P_{or}$ is the $<vagina, vaginitis>$ pair. We have analyzed this error in the previous discussion where we thought that a threshold would be of help. In our informal experiment, we apply a threshold of 6 for the number of hits returned for each query pattern. As result, in the last iteration, the precision was increased to 1 but the recall was decreased to 0.67. To get the balance between the precision and recall values, our best practice is to turn the threshold off.

4.6 Extracting Abbreviations

4.6.1 Previous Work

A pattern-based approach has also been used to recognize abbreviations and acronyms. Yu et al. (2002) developed the AbbRE system that recognizes parenthetical expressions such as,

(21) “<abbreviation> (<long form>)” or “<long form> (<abbreviation>)”

and then use a set of rules, which are obtained from common conventions for creating abbreviations, to map abbreviations to their long forms.

The majority of errors are related to the selection of the long forms, especially when the input is raw text. To overcome this problem, Pustejovsky et al. (2001) use a shallow parsed text as input, and add syntactic constrains into their patterns. One of the most frequent pattern in their experiment is,
\[(22) \quad < T_{\text{LF}} \cdot \text{NP}_1 > < (>)< T_{\text{A}} \cdot \text{NP}_2 > < ) >\]

where \( T_{\text{LF}} \) is a long form target, \( T_{\text{A}} \) is an acronym target, and \( \text{NP} \) is a noun phrase. In other experiments, the authors add new patterns which significantly increase recall with a small detriment in precision.

A more recent study on identifying abbreviation definitions in biomedical text is reported in Schwart and Hearst (2003). The authors proposed a simple and fast algorithm to identify \(<\text{“short form”}, \text{“long form”}\>\) pairs, whose problems have been addressed in Pustejovsky et al. (2001) above. Besides identifying abbreviations that follow a predictable pattern, where each letter in the short form corresponds to the first letter of each word in the long form, their algorithm also considers the cases where matching short form and long form requires skipping some words in the long form, as in \textit{Gcn5-related N-acetyltransferase (GNAT)}.

Another issue which has not been addressed in the previous studies is multilingual long form. In non-English documents such as documents in Dutch, we may find an abbreviation pair whose short form follows its form in English (e.g., \textit{WHO}), and whose long form is in Dutch (e.g., \textit{Wereldgezondheidsoorganisatie ‘World Health Organization’}). Obviously, we will not be able to find any matching in the pair if we use above approaches.

4.6.2 Method

Unlike synonym variation whose arguments are related in meaning, abbreviation variation (also acronym variation) have arguments which are related in forms, i.e., an argument is a short or a long form of the other argument. For example, \textit{VN} is an abbreviation or a short form of \textit{Verenigde Naties ‘United Nations’}. Inspired by this characteristic, our approach to extracting abbreviations will be based on common conventions in the creation of abbreviations. And since we work with Dutch documents, we also consider the multilingual issue.

---

**Figure 4.3**: The diagram of the abbreviation extraction method.

The diagram of our abbreviation extraction method is shown in Figure 4.3. Given input texts annotated with terms (Section 4.3), the method uses the following two main steps to extract abbreviation pairs:
### Table 4.7: Regular expressions for identifying abbreviations.

<table>
<thead>
<tr>
<th>Regular expression</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>/[A-Z]{2,} /</td>
<td>HIV</td>
</tr>
<tr>
<td>/[A-Z][a-z]???[A-Z]/</td>
<td>IgE, Ph.D, UV-B</td>
</tr>
<tr>
<td>/&quot;[A-Z][a-z]$/</td>
<td>Ig</td>
</tr>
<tr>
<td>/&quot;[A-Z][A-Z]([A-Z])*/</td>
<td>K.U</td>
</tr>
<tr>
<td>/&quot;[a-z][A-Z]+$/</td>
<td>mV</td>
</tr>
</tbody>
</table>

**Step 1** For each annotated input sentence, the *Tuple Extraction* process searches for the occurrence of abbreviations by applying regular expressions in Table 4.7 to all of the terms in the sentence. For each identified abbreviation, the component generates two tuples as follows:

- `<term1, abbreviation>`
- `<abbreviation, term2>`

where *term1* is the nearest term occurring before the abbreviation, and *term2* is the nearest term occurring after the abbreviation. These terms are the candidate long forms of the abbreviation.

**Step 2** For each tuple, the *Abbreviation Matching* process determines whether the long form argument matches with the abbreviation argument. For this evaluation, we use a set of pattern-matching rules described in Section 4.6.3. Only tuples that match the rules will be returned as abbreviation pairs.

Unlike the methods to detect synonym pairs, this method does not need any seed and pattern to identify the occurrence of new abbreviation tuples. Instead, it analyzes all sentences in one run to get the occurrence of abbreviations and their candidate long forms. As an example, consider the following input sentence annotated with terms:

(23)  De/det <meeste/adj mensen/noun> moeten/verb deelnemen/verb aan/prep een/det <officieel/adj trainingsprogramma/noun> om/comp te/comp stoppen/verb met/prep <drinken/noun> ,/punct zoals/comp die/det van/prep de/det <AA/name> (/punct <Anonieme/name Alcoholisten/name>) //punct ./punct

Most people should participate in a formal training program to stop drinking, such as the AA (Anonymous Alcoholics).

The sentence contains one abbreviation, i.e. **AA**, which is located between two nearest terms, i.e. *drinken* ‘drinking’ and *Anonieme Alcoholisten* ‘Anonymous Alcoholic’. Two candidate tuples will be generated for this abbreviation, namely:

- `<drinken, AA>`
- `<AA, Anonieme Alcoholisten>`

In the second step, the first tuple will be discarded since its long form (*drinken*) does not match with its abbreviation (**AA**). On the other hand, the second tuple has a long form (**Anonieme Alcoholisten**) that matches with the
<table>
<thead>
<tr>
<th>No</th>
<th>Rule</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The abbreviation matches the first letter of each words in the long form (LF).</td>
<td><em>Algemene Wet Bijzondere Ziektekosten (AWBZ)</em></td>
</tr>
<tr>
<td>2</td>
<td>The middle letters of an abbr. can be skipped if the first and the last letters match the first and the last words of the LF.</td>
<td><em>Rijksuniversiteit Groningen (RUG)</em>, <em>Immunoglobine E (IgE)</em></td>
</tr>
<tr>
<td>3</td>
<td>The abbreviation letters match middle letters of the compound word LF if the first letter of the LF matches the first letter of the abbreviation.</td>
<td><em>Wereldegezondheidsorganisatie (WGO)</em>, <em>carbon monoxide (CO)</em></td>
</tr>
<tr>
<td>4</td>
<td>If the first letter of an abbreviation does not match the first letter of the LF, translate the LF, and consider as a synonym if any of the above rules holds.</td>
<td><em>stikstofmonoxide (NO)</em>, <em>→ nitric oxide (NO)</em>, <em>koolmonoxide (CO)</em>, <em>→ carbon monoxide (CO)</em></td>
</tr>
<tr>
<td>5</td>
<td>If the first letter of an abbreviation does not match the first letter of the LF, discard the first word of LF, and start the evaluation with Rule 1 again.</td>
<td><em>verschillende Centra Algemeen Welzijn (CAW)</em></td>
</tr>
<tr>
<td>6</td>
<td>If both arguments have characteristics of an abbreviation, consider the pair as a synonym pair.</td>
<td><em>MDMA (XTC)</em></td>
</tr>
<tr>
<td>7</td>
<td>The above rules are iteratively applied in thier order until the abbreviation is completely matched or no match found.</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4.8: Pattern-matching rules for mapping an abbreviation to its long form.

abbreviation, and therefore it will be returned as an abbreviation pair. Thus, the matching value of each tuple is defined by the following equation:

\[
\text{Match}(x, y) = \begin{cases} 
  1 & \text{if } x \text{ matches } y \text{ according to rules in Table 4.8} \\
  0 & \text{otherwise}
\end{cases} \tag{4.7}
\]

where \(x\) is an abbreviation and \(y\) is its long form.

To evaluate the extracted pairs, the same evaluation technique as in Section 4.5.3, where a set of phrase queries are submitted to the search engine, can also be applied for this task. However, based on our informal experiments, the extracted abbreviation pairs have already shown a good performance. Nevertheless, in the evaluation section, we will compare the performance of these methods.

4.6.3 Matching Abbreviation Pairs

We now describe our abbreviation matching technique in Step 2 above. Since the creation of abbreviations follow some common requirements, we use a set of rules in Table 4.8 which reflect the requirements to identify relevant abbreviations and their long forms.

Rules 1-3 are adapted from Yu et al. (2002), while rules 4-6 are our additions. Rule 4 is created for English abbreviations whose long forms are in Dutch, such
as the pair $<\text{NO}, \text{stikstofmonoxide}>$. To evaluate this pair, first, we translate its long form into English which is resulted in nitric oxide, and then repeat the evaluation from Rule 1.

Rule 5 concerns the problem of the selection of long forms where a relevant long form may reside within a longer string. This rule allows the extraction of a relevant substring from a longer string, such as the long form Centra Algemeen Welzijn ‘General Welfare Centers’ from the string verschillende Centra Algemeen Welzijn ‘various General Welfare Centers’.

Our last addition is Rule 6, which is based on the occurrence of $<\text{abbreviation}, \text{abbreviation}>$ pairs in the corpus. For example, the pair $<\text{MDMA}, \text{XTC}>$ has abbreviations in both of its arguments which refer to the same concept. Actually, this abbreviation pair has also been extracted during the extraction of synonyms. However, since this pair is captured during the extraction of abbreviations, we will flag the pair as a synonym pair.

The rules in the table should be evaluated in order. First, an abbreviation pair is evaluated by Rule 1. If this rule does not find any match, the pair will be evaluated in the next rule until a matching is found ($\text{Match}(x, y) = 1$) or until the last rule is elapsed without finding any matching ($\text{Match}(x, y) = 0$).

### 4.6.4 Experiment and Results

We evaluate the abbreviation extraction method on the same corpus as used in the synonym extraction experiment, which contains 57,004 parsed sentences. In the first step, we identify short forms of abbreviations using regular expressions in Table 4.7, and then extract the abbreviations if there are preceding or succeeding candidate terms. From this step, we get 832 abbreviation-definition pairs.

In the next step, we apply the pattern-matching rules in Table 4.8 to identify correct pairs of short forms and long forms. From this step we get 82 abbreviation-definition pairs which are labeled as correct by the rules. Some examples of the identified abbreviation pairs are provided in Table 4.9. This table shows that, in general, the pattern-matching rules can identify abbreviation pairs that contain correct mappings of abbreviations and their long forms. Moreover, the rules are able to recognize abbreviation pairs that contain short forms in English/Latin and long forms in Dutch, such as $<\text{CO}_2, \text{kooldioxide}>$. This example is apparently also a chemical formula ($\text{CO}_2$), which can also be detected using a specialized method such as reported in Sun et al. (2007). However, the rule which supports a translation also introduces errors as shown by the abbreviation pair $<\text{RSI, relatie}>$. Its translated long form, relationship, apparently also matches with its abbreviation, RSI.

The problem in the selection of long forms is a common problem in the abbreviation extraction task. This problem is shown by the pair $<\text{JCA, naam juveniele chronische artritis}>$, whose argument contains an incorrect long form. If a correct long form is a substring of a longer string, our pattern-matching rules will be able to identify the relevant substring. For that example, our method can extract the substring juveniele chronische artritis from its longer string. However, if the argument contains a corrupted long form, which could be caused by PoS-tagging errors, the pattern-matching rules will not able to get the relevant long form and will discard the pair. For example, the pair $<\text{BERA, electrical response audiometry}>$ (not in the table) was discarded because it con-
<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Long form</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>Anonieme Alcoholisten</td>
</tr>
<tr>
<td>ADH</td>
<td>antidiuretisch hormoon</td>
</tr>
<tr>
<td>ASD</td>
<td>atriumseptumdefect</td>
</tr>
<tr>
<td>ASRM</td>
<td>American Society for Reproductive Medicine</td>
</tr>
<tr>
<td>BMI</td>
<td>Body Mass Index</td>
</tr>
<tr>
<td>BUN</td>
<td>blood urea nitrogen</td>
</tr>
<tr>
<td>CAPD</td>
<td>continue ambulante peritoneale dialyse</td>
</tr>
<tr>
<td>CHT</td>
<td>congenitale hypothyreödie</td>
</tr>
<tr>
<td>CMV</td>
<td>cytomegalovirus</td>
</tr>
<tr>
<td>CO2</td>
<td>kooldioxide ⇒ carbon dioxide</td>
</tr>
<tr>
<td>CT</td>
<td>computertomografie</td>
</tr>
<tr>
<td>CVA</td>
<td>cerebrovasculair accident</td>
</tr>
<tr>
<td>DIC</td>
<td>disseminated intravascular coagulation</td>
</tr>
<tr>
<td>EBV</td>
<td>Epstein-Barr-virus</td>
</tr>
<tr>
<td>ECG</td>
<td>elektrocardiografie</td>
</tr>
<tr>
<td>EDTA</td>
<td>EthyleenDiaminoTetra-Azijnzuur</td>
</tr>
<tr>
<td>EEG</td>
<td>elektro-encefalogram</td>
</tr>
<tr>
<td>ERCP</td>
<td>endoscopische retrograde cholangiopancreatografie</td>
</tr>
<tr>
<td>GIFT</td>
<td>gamete intrafallopian transfer</td>
</tr>
<tr>
<td>HCV</td>
<td>hepatitis C-virus</td>
</tr>
<tr>
<td>HIV</td>
<td>humaan-immunodeficiëntievirus</td>
</tr>
<tr>
<td>HPV</td>
<td>humaan papillomavirus</td>
</tr>
<tr>
<td>IADM</td>
<td>insulinafhankelijke diabetes mellitus</td>
</tr>
<tr>
<td>INH</td>
<td>isonicotinezuurhydrazide</td>
</tr>
<tr>
<td>IVF</td>
<td>in-vitrofertilisatie</td>
</tr>
<tr>
<td>JCA</td>
<td>naam juveniele chronische artritis ⇒ juveniele chronische artritis</td>
</tr>
<tr>
<td>LAAM</td>
<td>L-alfa-acetylmethadol</td>
</tr>
<tr>
<td>LSD</td>
<td>lysergeenzuurdiethylamide</td>
</tr>
<tr>
<td>LTH</td>
<td>luteotroop hormoon</td>
</tr>
<tr>
<td>MBD</td>
<td>afk. v. Eng. Minimal Brain Damage ⇒ Minimal Brain Damage</td>
</tr>
<tr>
<td>ME</td>
<td>mannen *</td>
</tr>
<tr>
<td>MRI</td>
<td>magnetic resonance imaging</td>
</tr>
<tr>
<td>MRI</td>
<td>magnetische kernspinnresonantie</td>
</tr>
<tr>
<td>MRI</td>
<td>magnetische resonantie</td>
</tr>
<tr>
<td>PSA</td>
<td>goedaardige prostaatvergroting ⇒ prostaatvergroting *</td>
</tr>
<tr>
<td>PSA</td>
<td>onbehandelde prostaatkanker ⇒ prostaatkanker *</td>
</tr>
<tr>
<td>PTCA</td>
<td>percutane transliminale coronaire angioplasty</td>
</tr>
<tr>
<td>RDS</td>
<td>respiratory distress syndrome</td>
</tr>
<tr>
<td>RSI</td>
<td>Repetitive Strain Injury</td>
</tr>
<tr>
<td>RS</td>
<td>relatie ⇒ relationship *</td>
</tr>
<tr>
<td>WHO</td>
<td>Wereldgezondheidsorganisatie</td>
</tr>
<tr>
<td>WRULD</td>
<td>Work Related Upper-Limb Disorders</td>
</tr>
</tbody>
</table>

Table 4.9: Some examples of abbreviations and their long forms extracted by using the pattern-matching rules. The asterisk (*) signs indicate incorrect pairs and the right arrow signs (⇒) indicate the correct matches detected from the long forms.
4.6. Extracting Abbreviations

Table 4.10: The performance of the abbreviation extraction methods.

<table>
<thead>
<tr>
<th>Setting</th>
<th>#Pairs</th>
<th>P</th>
<th>R</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>P&lt;sub&gt;or&lt;/sub&gt;</td>
<td>65</td>
<td>0.96</td>
<td>0.54</td>
<td>0.69</td>
</tr>
<tr>
<td>P&lt;sub&gt;N&lt;/sub&gt;EAR</td>
<td>434</td>
<td>0.66</td>
<td>0.80</td>
<td>0.72</td>
</tr>
<tr>
<td>Rules</td>
<td>82</td>
<td>0.98</td>
<td>0.84</td>
<td>0.90</td>
</tr>
<tr>
<td>Rules + P&lt;sub&gt;or&lt;/sub&gt;</td>
<td>109</td>
<td>0.96</td>
<td>0.88</td>
<td>0.92</td>
</tr>
</tbody>
</table>

To evaluate the abbreviation extraction method, we compare the technique which uses the matching rules in Section 4.6 (Rules) and the one which uses the query patterns in Section 4.5.3 (P<sub>or</sub> and P<sub>N</sub>EAR). In order to measure their precision, recall, and F-measure values, we randomly select 100 abbreviations occurring in the corpus and manually evaluate their long forms, which consist of 50 expected true positive pairs (E<sup>+</sup>) and 50 expected true negative pairs (E<sup>-</sup>). And to get a set of observed abbreviation-definition pairs (O) for each experiment setting, which count the number of its abbreviation-definition pairs which overlap with E<sup>+</sup> (to get O<sup>+</sup> value) and with E<sup>-</sup> (to get O<sup>-</sup> value).

Table 4.10 shows the numbers of the extracted abbreviation pairs and the performance at four extraction settings, i.e., P<sub>or</sub>, P<sub>N</sub>EAR, Rules, and a combination between Rules and P<sub>or</sub> methods. In the last setting, we label an abbreviation-definition pair as correct if at least one of the combining methods classifies it as correct.

Compared to the P<sub>N</sub>EAR setting, the P<sub>or</sub> and Rules settings are more restrictive with higher precision. These characteristics are shown by the smaller numbers of extracted abbreviation pairs at the P<sub>or</sub> and Rules settings and their higher performance compared to those of the P<sub>N</sub>EAR setting. The performance of the query patterns for this task is also consistent with their performance for the previous synonym extraction task (see also Table 4.6). In the present experiment, P<sub>or</sub> outperforms P<sub>N</sub>EAR in precision (0.96 v.s. 0.66) and, on the other hand, P<sub>N</sub>EAR outperforms P<sub>or</sub> in recall (0.80 v.s. 0.54). However, these patterns cannot beat Rules in both precision (0.98) and recall (0.84).

The high performance of the Rules setting compared to the P<sub>or</sub> and P<sub>N</sub>EAR settings is caused by the fact that abbreviations and their long forms can be easily recognized only from their formations. On the other hand, the abbreviation pairs may contain new abbreviations which do not occur in the Web pages. Thus, the last two settings may suffer from low recall. However, if one of these settings, i.e. P<sub>or</sub>, is combined with the Rules setting, we get the best recall (0.88) and F-measure values (0.92), although the precision is slightly decreased (0.96). This improvement is contributed by the addition of abbreviation pairs which do not match with the Rules. For example, the Rules setting misses the pairs <PABA, para-aminobenzoëzuur> and <HiB, Haemophilus influenzae type b> which are successfully recognized by P<sub>or</sub>.

tains a corrupted long form, whose correct one is brain stem electrical response audiometry. The word stem in this long form was tagged as a verb by the Alpino, which caused the linguistic filter to miss the whole string.
4.7 Summary

This chapter presents our investigations to answer the fourth research question on the types of term variation and on the methods to extract the variation types in Dutch documents. In section 4.2, we investigate term variation which is frequently found in the medical QA system, and then we describe our methods to extract the variation from medical corpora in Section 4.5 and 4.6. From medical questions collected from the Internet, we found three types of term variation, namely, synonymy, abbreviation, and grammar based variation. Only the first two types of variation are extracted in this chapter.

Our method to extract synonym variation is mainly based on the DIPRE method (Brin, 1999). First, using a seed list and a set of generated patterns we extract synonym tuples from the corpus. Then, we apply the tuple evaluation method described in (Lin et al., 2003), which submits several phrase queries to a Web search engine, to evaluate the compatibility of the synonym tuples which result in a set of synonym pairs. Since the size of our corpus is relatively small, we repeat the process in several iterations by using the extracted synonym pairs as a new seed list for the next iteration. This approach resulted in a high precision (0.98) with a high F-measure (0.82) for the strict query patterns $P_{\text{or}}$, and a high recall (0.85) but with a lower F-measure (0.72) for the more open query patterns $P_{\text{N E A R}}$. It is better to provide more synonym pairs as the initial seed list (e.g., around 20 synonym pairs selected manually from the corpus), since this will result in a high recall with few iterations.

Previous methods to extract abbreviations have achieved high precision and recall. Therefore, for this experiment we adopt one of the existing methods and adapt it for our implementation. To extract abbreviation variation, first, we use a linguistic filter to select abbreviations and their long forms. We assume that both forms are terminological forms, therefore, they can be detected using a term extraction technique. Then, we evaluate if an abbreviation and its neighboring term develop an abbreviation pair using a method as developed in Yu et al. (2002), which is based on pattern-matching rules. We add new rules to accommodate pairs whose arguments are of different languages and to select correct long forms from incorrectly extracted ones. We compare this method with the previous technique of evaluating synonym pairs using the search engine, which resulted in high precision (0.98) and recall (0.84) for simple pattern-matching rules.