Identifying SNOMED Clinical Terms in EHR: Full-Text Search vs. Word Embedding

Peter Fogh, Joachim Klokkervoll, Samuel Nygaard,
Mike Pedersen, Lynge Poulsgaard and Christian Slot

{pfogh12, jklokk12, snpe12, mipede12, lkpo12, cslot12}@student.aau.dk

December 18, 2015

Aalborg University

Department of Computer Science

Abstract

Mapping between medical systems is a complex problem. The electronic health record system of Central Denmark Region contains many free-text entries and as such could benefit from mapping these into a more structured system. This paper explores two different methods for automatically mapping the current entries to a new system. The two methods used, doc2vec and full-text search, are word embedding and text retrieval algorithms, respectively. These methods have been evaluated using a manually mapped verification set of 382 free-text fragments mapped to corresponding SNOMED CT concepts, and have been compared to a naive mapper, showing positive results. Customizing the full-text search method yielded the best result, with a correct mapping for 17.3 % of the verification set.

Aalborg, December 18, 2015

Peter Fogh
Student ID: 20124274
E-mail: pfogh12@student.aau.dk

Joachim Klokkervoll
Student ID: 20124277
E-mail: jklokk12@student.aau.dk

Samuel Nygaard
Student ID: 20124278
E-mail: snpe12@student.aau.dk

Mike Pedersen
Student ID: 20124283
E-mail: mipede12@student.aau.dk

Lynge Poulsgaard
Student ID: 20124272
E-mail: lkpo12@student.aau.dk

Christian Slot
Student ID: 20127075
E-mail: cslot12@student.aau.dk

1 Introduction

Medical data in Denmark, including health records, has undergone a change from physical reports and documents in favor of digital data, i.e. electronic health records (EHR). The primary purpose of EHR is to document the treatment of a patient, and therefore provide useful information for subsequent treatments. An advantage of EHRs is the opportunity for secondary use such as analysis, reporting, and comparison.

The data sets explored in this paper are provided by the Central Denmark Region[1] (henceforth "the Region"). The Region is collaborating with the intention to examine the opportunities of mapping their EHRs to the international medical terminology Systematized Nomenclature of Medicine – Clinical Terms (SNOMED CT) [1]. The data in their EHR system is to some degree structured, but is generally in the form of unstructured free-text, which complicates secondary use.

SNOMED CT supports multiple languages and encapsulates both a clinical classification system and a research terminology to ensure semantic interoperability. This makes it beneficial to investigate the possibilities of mapping free-text from unstructured and non-standardized EHRs to the internationally standardized SNOMED CT.

This paper explores solutions to the problem:

Computing a reliable mapping from free-text in Danish EHRs to SNOMED CT concepts.

From the field of natural language processing (NLP), this paper investigates two different methods for solving this mapping problem: Text retrieval and word embedding. Text retrieval is implemented using full-text search (FTS) in SQL Server [2], while the word embedding method is implemented using doc2vec [3] provided by the Gensim library [4]. The paper includes a description of the tools, their implementation, and a final evaluation and comparison of their results.

1.1 Related work

Various other work has been done, regarding an implementation of SNOMED CT in existing EHR systems. Holm et al. [5] of Project SNOMED Klurig in Östergötland, Sweden presented their findings after implementing SNOMED CT in a medical environment similar to the ones used in Denmark. The presentation concluded that the implementation of mapping an EHR system would benefit from the EHR data already being structured. However, in this paper we will focus on mapping unstructured EHR data, i.e. natural language.

The paper by Peng et al. [6] has shown that language understanding can be achieved by using recurrent neural networks. This method is useful for finding the semantic meaning of the words in a sentence, which relates to finding the correct SNOMED CT terminology for a given word.

Stenzhorn et al. [7] have investigated the possibilities of using the natural language tools OpenNLP and MorphoSaurus for mapping medical text to SNOMED CT concepts. This paper follows a similar approach, using different natural language tools from the Gensim library.

2 Problem definition

The fundamental issues explored in this paper are those that arise with the mapping of free-text in EHRs to SNOMED CT concepts. This problem has been proposed by the Region in collaboration with the Danish data analytics company Enversion A/S[2]. Enversion, who specializes in analyzing medical data, and the Region imagine that the patients living in the Central Denmark Region would benefit from their EHRs being mapped into the SNOMED CT terminology. This section will explain the details and aspects of the problem.

2.1 SNOMED CT

The International Health Terminology Standards Development Organisation (IHTSDO) owns and maintains SNOMED CT: *"...the most comprehensive and precise clinical health terminology product in the world ..."* [8]. The purpose of SNOMED CT is to encapsulate both clinical and scientific terminologies. This makes semantic interoperability possible, i.e. two different computer systems can exchange data unambiguously.

As described by IHTSDO [1], SNOMED CT consist of three main components: *Concepts*, *descriptions,* and *relationships,* illustrated in Figure 1. Concepts and relations in SNOMED CT are generally denoted with a leading and trailing '|', e.g. the |is a| relation.

Concepts are representations of unique clinical meanings and are referenced by a unique numeric SNOMED CT identifier, i.e. an eight digit number.

Descriptions are human readable terms attached to every concept, there are two types of descriptions: Fully specified name (FSN) and synonyms. The FSN is the unambiguous description of a concept. Each supported language also has a preferred synonym for most of the concepts and may have some acceptable synonyms as well. As of today, SNOMED CT has been translated into a number of different languages, including Danish, these translations supply synonyms that refer to SNOMED CT concepts.

Relationships are directional unions between two concepts. Relationships are either attribute relations or the subtype relationship called the |is a| relation. All concepts, with the exception of the root |SNOMED CT Concept|, has one or more |is a| relations. These |is a| relations forms a directed acyclic graph (DAG) that resembles a tree structure with the most specific concepts inheriting from the concepts above up

[1] http://rm.dk

[2] http://enversion.dk

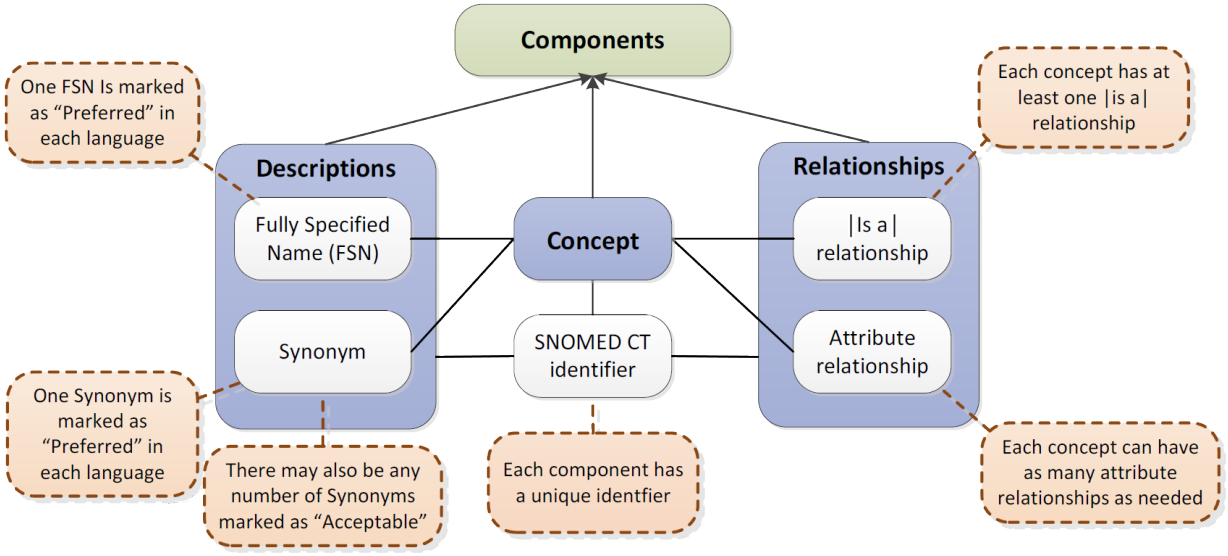

Figure 1: SNOMED CT's structure [1]

to the root concept. Attribute relations are combining concepts with additional defining characteristics, e.g. finding site of a disorder.

These components and their structure enables SNOMED CT to describe and classify medical terms with fine granularity. This is a big advantage as documenting EHR in SNOMED CT is possible with high level of detail. However, this granularity might also become a problem when mapping existing terminologies, or free-text, to SNOMED CT as the level of detail could make it difficult to identify the correct description with certainty.

SNOMED CT consists of more core capabilities which are not used in this paper and therefore not explained further.

2.2 Data sets

This section will describe the different data sets used in this research and is concerned with the source set of the EHRs and the destination set consisting of the Danish SNOMED CT descriptions.

2.2.1 EHR free-text data

An *activity* is a documented event regarding treatment, examination, operation, consultation, etc. The source data consists of free-text entries from the EHRs provided by the Region and contains almost 1.4 million activity entries. The average character length of these free-text entries is 160 characters, varying from 0 to 27,379 characters.

An activity can have several elements, meaning that the documentation of an activity is divided into several textfields. An example of fabricated EHR free-text entries is shown in Table 1. The first two rows have the same **Activity ID**, but the first element is the textual representation of a blood pressure measurement and the second element is a comment regarding the measurement from the first element. If there exists only one element for a given activity it is often a journal note or a single comment regarding the treatment of the patient.

2.2.2 Descriptions

The destination set for the mapping is the descriptions from the Danish extension of SNOMED CT. An example showing the SNOMED CT descriptions of the concept |African trypanosomiasis (disorder)| can be seen in Table 2. In this example, there are several English descriptions for the concept but only one Danish description. In the SNOMED CT RF2 Release as of 2015-07-31, there is a total of 830,413 English descriptions and 256,066 Danish descriptions.

There are two major problems with the set of Danish descriptions: First, out of 317,364 active concepts there exist 61,636 concepts without a Danish description, and second, only 291 concepts have more than one Danish description. This makes the mapping problem from Danish text to SNOMED CT concepts more difficult than mapping English text.

2.3 Identifying descriptions in free-text

In this section we will discuss the subproblems that emerge from the main problem when trying to map free-text entries to descriptions.

2.3.1 Ambiguous mapping

The first subproblem is that of ambiguous mapping caused by the high level of detail in SNOMED CT. This means that one free-text can map to arbitrarily many different concepts that have similar descriptions. An example of this is shown in Table 3. In the table it is shown that the free-text with **Map ID** 2 can meaningfully map to both |Abnormal blood pressure (finding)| and |Hypertension (disorder)|, and it is necessary to understand the context of the free-text to map correctly. However the free-text

Activity ID	Element	Data type	Free-text value
321809032	1	string	180/120
321809032	2	string	Abnormal high blood pressure
351690241	1	string	The patient suffers from African sleeping sickness

Table 1: Fabricated example of EHR activity entries

Description ID	Concept ID	Language	Type	Description
757552012	27031003	English	FSN	African trypanosomiasis (disorder)
45215013	27031003	English	SYN	African sleeping sickness
45217017	27031003	English	SYN	Sleeping sickness
19349515118	27031003	Danish	SYN	Afrikansk sovesyge

Table 2: Example of SNOMED CT descriptions for the concept |African trypanosomiasis (disorder)|

The patient suffers from African sleeping sickness but does not have abnormal high blood pressure (120/80) or fast pulse.

Figure 2: Fabricated example of an EHR free-text entry

with **Map ID** 3 is conveniently mapped to the disorder |African trypanosomiasis| because the free-text is specifically worded and the phrase "African sleeping sickness" does not appear often in the SNOMED CT descriptions.

2.3.2 Multiple mappings

The second subproblem with identifying descriptions in free-text is that a sentence often describes several SNOMED CT concepts. This means that a sentence can contain many concepts, e.g. a journal note might contain the text in Figure 2. In such cases the optimal output would be identifying all three underlined concepts contained in the sentence.

2.3.3 Identifying negation

A third subproblem regarding EHR data is to identify the polarity throughout a free-text. An example of this is seen in the free-text entry from Figure 2, that contains three concepts where the last two should have a negative polarity as the word "not" binds to both of them. If a solution extracts the three underlined concepts in a naive way, the negation context for some concepts might be lost.

2.3.4 Null mapping

The fourth subproblem concerns the fact that the SNOMED CT terminology does not support free-text with the lowest granularity, like numeric values or personal names, thus some free-text values should not be mapped to a SNOMED CT concept, henceforth called a null mapping. An example of this is also shown in Table 3, where the mapping with **Map ID** 1 has the text "180/120", denoting the numeric measure of the blood pressure. Understand the meaning of such numeric values will require additional context.

2.4 Output

With focus on the main problem, four choices for solving the aforementioned subproblems have been made.

The ambiguous mapping problem can be solved by finding a measure for choosing the best mapping and then choosing the mapping with the best measure.

If such a measure exists, then it would make sense to choose a threshold for this measure. As such, any mapping with a measure lower than a certain threshold will become a null mapping. Preferably, all entries that would otherwise result in a null mapping should have a measure lower than the threshold, while all mappings resulting in a correct mapping should have a higher measure.

Prior to mapping a free-text entry, it is split into fragments. If done in a proper way, this deals with both the multiple mappings problem and the negation problem.

2.4.1 Desired output

The desired mapping would be to combine solutions for the choices above into an unambiguous one-to-many mapping where the free-text given as input is split into fragments, flagged with the right polarity, and possibly mapped to a concept. The expected output from mapping the previously used free-text examples to SNOMED CT concepts can be seen in Table 4.

Fragment text	Concept ID	Polarity
The patient suffers from African sleeping sickness	27031003	Positive
but does not have abnormal high blood pressure	38936003	Negative
(120/80)	-	-
or fast pulse.	86651002	Negative

Table 4: Expected output for the input from Figure 2

3 Background

The following section contains presentations of the necessary background knowledge needed for understanding the solutions presented in this paper.

Map ID	Free-text value	Concept ID	SNOMED CT FSN	Matched description
1	180/120	-	-	-
2	Abnormal high blood pressure	38936003	Abnormal blood pressure (finding)	Abnormal blood pressure
		38341003	Hypertensive disorder, systemic arterial (disorder)	High blood pressure
3	The patient suffers from African sleeping sickness	27031003	African trypanosomiasis (disorder)	African sleeping sickness

Table 3: Example of the problem with mapping free-text values to the correct descriptions

3.1 Natural language processing

NLP is a large subject with many subcategories. In this project, the main goal of the NLP is information extraction. Information extraction means extracting relevant data from free-text and parsing it into structured data. These methods might be useful when trying to solve the multiple mappings problem stated in Subsection 2.3.2. The following section is based on the work by Nadkarni et al. [9], Jurafsky and Manning [10].

Originally, in the 1960's, natural language processing problems were solved via handwritten rules consisting more or less of complicated if-else chains or decision trees. It then evolved into using regular expressions and later machine learning algorithms.

Regular expressions Regular expressions are a way of defining search patterns for text search. This is the general approach used in NLP today and can be used as a solution for NLP or as a baseline for machine learning algorithms.

Machine learning Machine learning is a relatively new approach to NLP and can get good results with big and varied training sets. The main advantage of using machine learning in NLP is that the algorithm automatically focuses on the most common cases, while creating handwritten rules and regular expressions for these common cases may not be obvious. Another advantage of machine learning algorithms is that they are better at handling unfamiliar input provided they are given additional data. The biggest problem with using machine learning in NLP is the risk of over fitting for certain inputs that are not frequent in the test data.

3.2 Full-text search

Full-text search is *text retrieval* of free-text entries, based on queries.

Microsoft has developed their own implementation, Full-Text Search, in SQL Server and Azure SQL Database [2]. To run full-text queries a full-text index needs to be created on the table to be queried. Only one full-text index can be created on a table but can include arbitrarily many character-based columns. As any database index, a full-text index makes certain search queries faster and works by populating and using an inverted word list.

Full-text queries can consist of any combination of the following search conditions as defined by Microsoft [2]:

- *Simple term*: One or more words or phrases.
- *Prefix term*: A word or a phrase beginning with a specific prefix.
- *Generation term*: Inflectional forms of a specific word.
- *Proximity term*: A word or a phrase close to other words or phrases.
- *Thesaurus*: Synonymous forms of a specific word.
- *Weighted term*: Words or phrases using weighted values.

Full-Text Search supports several languages, including Danish. For each supported language, SQL Server provides linguistic components that support indexing and querying full-text data that is stored in that language. The linguistic components include the following:

Word breakers and stemmers A word breaker identifies word boundaries based on the lexical rules of a given language. Each word breaker is associated with a stemmer that conjugates verbs, e.g. the words *organizes* and *organizing* can be interpreted in the same way as their common base word *organize*. During index creation the word breaker and stemmer is used to structure the index. It is also possible to use word breaking and stemming in full-text queries.

Stoplists A stoplist is used during index creation to discard commonly occurring words that does not provide any context or meaning for searches. The discarded strings are called stopwords or noise words.

Thesaurus Full-text queries can search for synonyms of user-specified terms through the use of a thesaurus. SQL Server does not define any default thesauruses, therefore these must be created by the server administrator.

3.2.1 Full-text ranking

As Microsoft describes [11], when using the full-text search functions `FREETEXTTABLE` and `CONTAINSTABLE`, the results are ranked. `FREETEXTTABLE` queries always use stemming and synonyms to broaden the result. `CONTAINSTABLE` queries search for strict occurrences of the query-text by default. Both functions can be expanded to include any combination of the search conditions.

The two SQL Server functions use different ranking algorithms:

`CONTAINSTABLE` uses a ranking algorithm based on a logarithmic relationship between the number of concepts matched and the total number of concepts in the index together with the length of the concept description. When expanding `CONTAINSTABLE` with the `ISABOUT` search condition the ranking algorithm is expanded with a Jaccard similarity coefficient [12].

`FREETEXTTABLE` uses a ranking algorithm based on the Okapi BM25 algorithm [13], a best matching ranking algorithm. Okapi BM25 matches on relevance between documents using the probabilistic relevance model.

3.3 Doc2vec

Word embedding is a set of language modeling methods wherein a vocabulary is mapped to vectors in a low dimensional space. Where words are discrete, the corresponding vector representation is continuous and can thus more easily be compared to other words.

Doc2Vec builds on the foundation of *Word2Vec* [14, 15]; a tool built by Google engineers that efficiently implements the *continuous bag of words* and *skip-gram* architectures for generating vector representations of words. This means that it can, given a large corpus of text, generate a set of vectors such that the vector for word α will be close to the vector for word β if α and β often co-occur closely in the text corpus. According to J.R. Firth's hypothesis of often co-occurring words having similar meaning [16], semantically similar words will thus be clustered together when given a sufficiently large data set.

The semantic distance between two words can be found using the cosine distance of the two corresponding vectors. Geometrically, this corresponds to the angle between two vectors. Engineers at Google trained a word2vec model on a large Google News data set and the resulting model placed semantically similar terms like "big" and "bigger" close together. [14]

These vector representations can be compared using cosine distance as described above, but another option is using them as inputs for other machine learning algorithms. As the resulting vectors are fixed-length, they are often more suitable for many machine learning methods than a variable-length input.

Doc2vec [3] or *phrase vectors* is a similar method based on word2vec, but builds vectors for entire collections of words (documents) in addition to building vectors for individual words. Doc2vec implements *distributed bag of words* (DBOW) and *distributed memory* (DM) architectures, which are document versions of the aforementioned word2vec architectures. Like word vectors, the distance between two documents can be found using cosine distance or the vector can be used as input for another machine learning algorithm.

When training the doc2vec model, it is necessary to consider how different hyperparameters might impact the model. For example, the number of the dimensions in the vector space will heavily influence the locations of word and document vectors. Likewise, the possibility of using either the DM or DBOW architecture can also have a major impact. Distributed memory is generally better as it also uses the structure of the words in the document, but it varies on a case by case basis.

3.4 SNOMED CT distance

To properly compare a mapping of EHR free-text to the concept best describing the free-text, a distance function between the mapped and correct concept would be useful.

A straightforward way of calculating the distance is to utilize the DAG structure of SNOMED CT, by counting the |is a| relations that separates the two concepts. However, this might not always result in a fair distance, since the hierarchy contains a varying level of granularity. An example could be the two concepts |Body structure| and |Organism|, which are both children of |SNOMED CT Concept| and two other concepts |Entire left foot| and |Entire right foot| which are both children of |Entire foot|. Even though the first two concepts are very different semantically compared to the last two, they would have the same distance as seen in Figure 5 where this distance measure is called $\text{dist}_{\text{simple}}$.

Garla and Brandt [17] evaluated different distance measures and found a distance measure based on intrinsic information content (IC) to best capture this semantic distance. The intrinsic IC for a concept c is based on its number of leaves, number of subsumers[3], and the total number of leaves in the SNOMED CT DAG, which is 224,443:

$$\text{IC}(c) = -\ln\left(\frac{\frac{|\text{leaves}(c)|}{|\text{subsumers}(c)|}+1}{\text{maxleaves}+1}\right)$$

The distance between two concepts can be calculated using the IC function on both concepts and the lowest common subsumer (lcs). These values are then used to calculate the IC distance:

$$\text{dist}_{\text{IC}}(c_1, c_2) = \text{IC}(c_1) + \text{IC}(c_2) - 2 \cdot \text{IC}(\text{lcs}(c_1, c_2))$$

The calculated dist_{IC} between |Body structure| and |Organism| and between |Entire left foot| and |Entire right foot|, seen in Figure 5, will therefore be 6.132 and 0.16 respectively, as seen in Figure 3 and Figure 4. We conclude that the dist_{IC} for comparing mapping to SNOMED CT is better than $\text{dist}_{\text{simple}}$, since it utilizes the varying level of granularity in SNOMED CT.

[3] The subsumers of a is the set containing a and all ancestors of a

$$\begin{aligned}\text{dist}_{\text{IC}}(\text{Body structure}, \text{Organism}) &= \text{IC}(\text{Body structure}) + \text{IC}(\text{Organism}) - 2 \cdot \text{IC}(\text{SNOMED CT concept}) \\ &= -\ln\left(\frac{\frac{170700199}{2}+1}{224443+1}\right) - \ln\left(\frac{\frac{25646}{2}+1}{224443+1}\right) + 2 \cdot \ln\left(\frac{\frac{224443}{1}+1}{224443+1}\right) \\ &= 6.132\end{aligned}$$

Figure 3: Calculated dist_{IC} of |clinical finding| and |organism|

$$\begin{aligned}\text{dist}_{\text{IC}}(\text{Entire left foot}, \text{Entire right foot}) &= \text{IC}(\text{Entire left foot}) + \text{IC}(\text{Entire right foot}) - 2 \cdot \text{IC}(\text{Entire foot}) \\ &= -\ln\left(\frac{\frac{0}{29}+1}{224443+1}\right) - \ln\left(\frac{\frac{0}{29}+1}{224443+1}\right) + 2 \cdot \ln\left(\frac{\frac{2}{24}+1}{224443+1}\right) \\ &= 0.16\end{aligned}$$

Figure 4: Calculated dist_{IC} of |Entire left foot| and |Entire right foot|

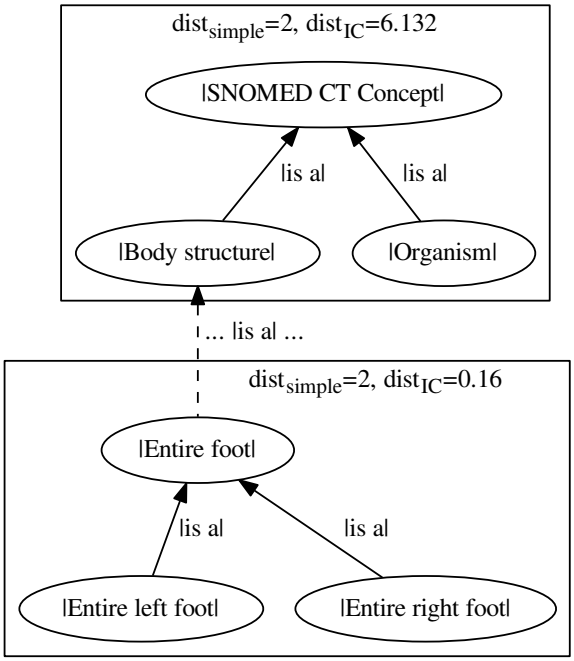

Figure 5: $\text{dist}_{\text{simple}}$ and dist_{IC} example

Type	Characters/words
Closed	but | so | . | ? | ! | -
Open	and | or | , | : | ;
Negation	no | not | none | nothing | neither

Table 5: Words and characters used for preprocessing, translated from Danish to English

4 Method

This section will describe the implementation of the solutions proposed in Subsection 2.4, based on the background knowledge described in Section 3.

4.1 Preprocessing

For solving the multiple mappings and negation problems described in Subsection 2.3, preprocessing the input free-text is necessary.

Subsection 3.1 describes two general methods for extracting information from free-text, namely regular expressions and machine learning. Due to the lack of Danish training data, NLP using regular expression is preferable to machine learning algorithms in this solution.

Using regular expressions to split the free-text into fragments, where each fragment carries only part of the full meaning, we can solve the multiple mappings problem, mentioned in Subsection 2.3.2, by mapping each fragment to a single concept.

Additionally, the identification of negation, described in Subsection 2.3.3, must be solved in a similar way. This can be done by flagging the current fragment with a polarity flag.

This idea of splitting the text into fragments have been achieved by splitting the free-text data on certain characters or words. A flag is set upon seeing so-called negation words. In this definition there are two types of splitting characters or words, open and closed. A negative polarity flag is repeated on fragments following a negation word as long as the seen splitting character between the fragments is of the open type. Closed type splitting resets the polarity when encountered. The different characters for open and closed splits along with the different matches for negation can be seen in Table 5.

4.2 Customizing full-text search

To support full-text searching, a full-text index has to be created on a table. This table should contain all the possible search results of a search query. As we want to find descriptions that match our text fragments, we have created a full-text index on a table containing these descriptions and the concepts that they describe.

To improve the results of full-text search we have tried

to customize its parameters and components in regards to our domain. These components are the stoplist, thesaurus, and ranking algorithm. While it is possible to alter the stoplist and thesaurus components directly, it is not possible to change the behavior of the ranking algorithm. Therefore, we have defined our own ranking function.

4.2.1 Custom stoplist

Default stoplists for the different languages are available after installing the full-text search module and contain the most used words in a language. Using a stoplist increases performance by filtering out irrelevant results when search queries contain these common words. This is a good starting point, but it is not domain specific. Analysis of the free-text data from the EHR source could provide domain specific stopwords. Saif et al. [18] proposes several methods for such an analysis. When applying their methods based on Zipf's law to our domain, they unveiled many words, such as the widely used word "pain", that might not be relevant for some searches. However, "pain" might provide a necessary context for other searches, and as such it should not be included in the stoplist. Because of this we have chosen not to add any domain specific stopwords to the stoplist.

4.2.2 Domain specific thesaurus

By extending the thesaurus with synonyms used in EHRs, it is possible to increase the amount of relevant, and maybe even more accurate, search results of full-text queries using medical terms. We have compiled such a list from a freely available medical synonym list provided by the Danish work organization FOA [19].

4.2.3 Ranking

Full-text search provides a ranking that can be used to order the results of a search query. However, for our domain it might not assign the highest ranking to the most relevant mapping. We see two reasons for this incorrect ordering.

First, the naming conventions of the descriptions creates a lot of short and similar documents in the full-text index. On such an index it is difficult for the ranking algorithm to distinguish between such descriptions. A consequence of this is that an exact match will be ranked lower than a match containing the fragment multiple times, e.g. a search for "blood" will result in both |Blood specimen from blood product (specimen)| and |Blood (substance)| being found, but rank the former higher than the latter.

Second, our fragments might contain irrelevant information for the search, but the ranking algorithm equally weights all words in a fragment, e.g. searching for "The patient talks about the present in past tense" will yield a higher rank for |Talks about the past| than the semantic meaning |Mixes past with preset| due to the search string containing the word "talk".

By defining our own custom ranking function that considers the specific domain of short and similar documents, a better ordering might be achieved. In this function we include the following measures and weights found through experimentation.

Full-text rank is the rank returned by the full-text search functions described is Subsection 3.2.1. These functions are also used to find all preliminary search result that we will calculate a custom rank for. The results returned by `CONTAINSTABLE` and `FREETEXTTABLE`, as well as their search rank, are processed by our custom ranking function.

Textual equality is a weight of two possible values. If the textual representation of the fragment is equal to the textual representation of the description, the weight is chosen to be 800. Otherwise this measure should not affect the custom rank and is assigned the weight 1. Full-text rank is higher for descriptions that contains the fragment more than once, but this weight will make sure that the exact matches will be ranked higher.

Shared substring is a weight of four possible values:

- **200** if the description is contained in the fragment.
- **160** if the fragment is a postfix of the description.
- **80** if it is a prefix of the description.
- **1** if none of the above cases are true.

The reasoning behind postfixes being ranked higher than prefixes, is that descriptions generally contain more context in the end of the description than the beginning. As an example, many concept descriptions contains "multiple sclerosis" where the specific type is specified as a prefix of multiple sclerosis, e.g. |Primary progressive multiple sclerosis (disorder)| and |Relapsing remitting multiple sclerosis (disorder)|.

Relative difference in text length is the absolute difference in text length relative to the longest text string. This relationship is multiplied with a weight of 20, and the result is lower bounded by 1. This weight will favor pairs of fragments and descriptions that are close to having the same length. This is to weigh down long descriptions for short fragments and vice versa.

$$\mathrm{RDTLW}(f,d) = \max\left(1, \frac{20\max(|f|,|d|)}{\max(1,(|f|-|d|)}\right)$$

Levenshtein distance is the edit distance between the fragment and the description [20]. For longer fragments and descriptions this measure should become less significant to avoid over-penalizing fragments with a lot of irrelevant information. Therefore we take the average length of the fragment and description and divide it by the Levenshtein distance (LevDist):

$$\mathrm{LDW}(f,d) = \frac{|f|+|d|}{2\max(1,\mathrm{LevDist}(f,d))}$$

These weights and measures are combined to create our custom ranking function seen in Figure 6. The function takes two parameters; a fragment f, and a description d.

$$\begin{aligned}\text{CustomRank}(f,d) &= \text{FTR}(f,d)\cdot\text{TEW}(f,d)\cdot\text{SSW}(f,d)\cdot\text{RDTLW}(f,d)\cdot\text{LDW}(f,d)\\ \text{where FTR}(f,d) &= \text{the full-text rank}\\ \text{TEW}(f,d) &= \text{the textual equality weight}\\ \text{SSW}(f,d) &= \text{the shared substring weight}\end{aligned}$$

Figure 6: Custom ranking function for full-text search

4.3 Doc2Vec

We utilize the Gensim library [4], an open-source topic modeling library which provides an implementation of both word2vec and doc2vec. All the decisions regarding the data, processing, or parameter tuning presented in this section will be evaluated in Subsection 5.4.

In Gensim, a document may either be tagged or untagged[4]. A tagged document gets an associated document vector. An untagged document has no associated document vector, and therefore cannot be inferred later. While untagged documents provide no direct influence on the document vectors, they do influence the underlying word vectors, which in turn influences the document vectors. Each description becomes a tagged document, tagged with the corresponding concept ID, and the remaining unstructured data becomes untagged documents.

The model is trained with three different data sets as can be seen in Table 6. Primarily, we train the doc2vec model with descriptions tagged with their corresponding concept. We also train the doc2vec model with untagged documents from EHRs and text from several Danish medical websites.

Data set	Description
SNO	SNOMED CT descriptions tagged with the concept ID
EHR	Untagged EHR notes
Web	Untagged scraped medicine websites

Table 6: Data sets used for training

Before training the models with the documents, we preprocess the data as described in Subsection 4.1. This can have either a negative or positive impact: The reduced strings have more semantically relevant words, but the less relevant words might have an impact for a sufficiently trained model. Given that we do not have a large amount of data for each concept, usually just a single document per concept, we hypothesize this preprocessing to be beneficial.

We train the model using *multiple-pass alpha-reduction*: The model is trained with the same data multiple times, linearly reducing the learning rate of the model as it progresses through the iterations. This is claimed by the authors of Gensim to produce better results for certain cases, as the individual documents will be trained multiple times with different learning rates. We shuffle the set of documents for each run, also based on a recommendation by the Gensim authors. [21]

To query the trained model for a concept to a string, we break the string into words, sum the corresponding word vectors, and find the most similar document vector. This approach has the downside of discarding the order of the words, which can have an important impact on the meaning of a document.

As we mentioned in Subsection 3.3, we could have used a more general machine learning algorithm than simply finding the most similar vector. Given that there is only a single document for each concept, we find it unlikely that a machine learning algorithm would have much predictive power, and thus the reason why we use a simpler approach based on finding the most similar vector. With additional data, using a more advanced machine learning algorithm would likely be beneficial.

[4] A document may have multiple tags, but this feature is not utilized in this paper.

5 Evaluation

The following section will evaluate the results of the two methods FTS and doc2vec. To evaluate the correctness of the mapping produced by the each method, a verification set have been manually created. Furthermore, two baseline methods are proposed for comparison.

5.1 Verification method

It cannot be concluded what the best configuration is for either of the two mapping techniques, based exclusively on previously described studies. Therefore, the purpose of this evaluation is not only to find which technique is best, but also to find the configuration for each technique that produces the performance. As such, each technique will be performed multiple times given the same input, i.e. the set of fragments. For each run only one parameter will be changed, thus providing a reliable comparison.

To compare the results of the two techniques and each of their runs, we use dist_{IC}, described in Subsection 3.4, to compute the mean distance between algorithmically mapped concepts and the concepts of the verification set. This mean distance will be used to measure the performance of an implementation.

5.1.1 Verification set

Thus we need a set of Danish fragments with the correct mapping to concepts. As it has not been possible to find such a data set, we have resorted to manually creating

our own. Using a subset of the EHR data mentioned in Subsection 2.2.1, we manually map 382 fragments to each of their corresponding concept IDs. As we, the authors, are in no circumstances experts on this subject, the correctness of the verification set is doubtful. A verification set built by experts would be preferable, but has not been obtainable for this paper.

5.1.2 Threshold value

We also need to consider that some fragments should not be mapped to any concept, i.e. the null mapping discussed in Subsection 2.3.4. Both techniques outputs a similarity measure in addition to the mapping itself. Assuming that there is a correlation between the reported similarity and the distance of the mapping, a threshold can be established for when the mapping of a fragment should be accepted or null mapped. If this threshold is set high, many false negatives (fragments that are incorrectly null mapped) will occur. If set too low, many false positives (fragments that are incorrectly mapped to a concept) will occur, in addition to the mapped concepts having a higher mean distance. Due to this trade-off, an optimal threshold depends on the desired outcome. To visualize this, we will present the number of false negatives, false positives, and mean distance as functions of the threshold value.

5.2 Baseline methods

To establish a baseline for comparison for our methods, we first consider a random mapper that simply chooses a random concept for a given string. We can calculate the mean distance of this mapper by taking the mean distance of all pairs of concepts. As this is $n^2 \approx \left(3.17 \cdot 10^5\right)^2 \approx 1.00 \cdot 10^{11}$ different pairs and thus becomes impractically large, we make do with sampling a subset. For a sample size of $n = 10000$ randomly chosen pairs, we found a mean distance of $m = 23.164$ and a standard deviation of $s = 2.916$. The expected standard error of the mean is thus $\frac{s}{\sqrt{n}} = 0.029$. Thus, 23.164 is a reasonable mean distance estimate for a random mapper.

We also consider a naive mapper that always maps to a single concept. The best naive mapper always maps to the concept which has the smallest sum of distances to all other concepts. This can be computed in $O(n^2)$ time. This is impractical to compute, but a likely contestant is the root concept. Let r be the root concept, as $\text{dist}_{\text{IC}}(c_1, c_2) = \text{IC}(c_1) + \text{IC}(c_2) - 2 \cdot \text{IC}(\text{lcs}(c_1, c_2))$, $\text{IC}(r) = 0$ and $\text{lcs}(r, c) = r$, then $\text{dist}_{\text{IC}}(r, c) = \text{IC}(c)$. We have found the average distance from the root to all other nodes to be 12.2.

5.3 Full-text search evaluation

In this section, we will determine the best configuration of full-text search for our domain and evaluate the results. We have looked at the following parameters:

Ranking algorithm is either our custom ranking function or the algorithm provided by default.

Stoplist is either used or not used when creating the full-text index on our description table.

Synonyms is either used or not used to expand search words.

As all three parameters can have two values there are 8 unique configurations. The result of running our verification set through each configuration can be seen in Table 7 and shows the fraction of direct hits and average distance, calculated using dist_{IC}. A direct hit is when the search returns the same concept as the verification set. It can be clearly seen that our custom algorithm performs better for our domain than the default ranking algorithm.

A scatter plot of the best configuration, Cust-stop-syn, can be seen in Figure 7. The plot shows the assigned rank and the distance to the correct concept for each fragment.

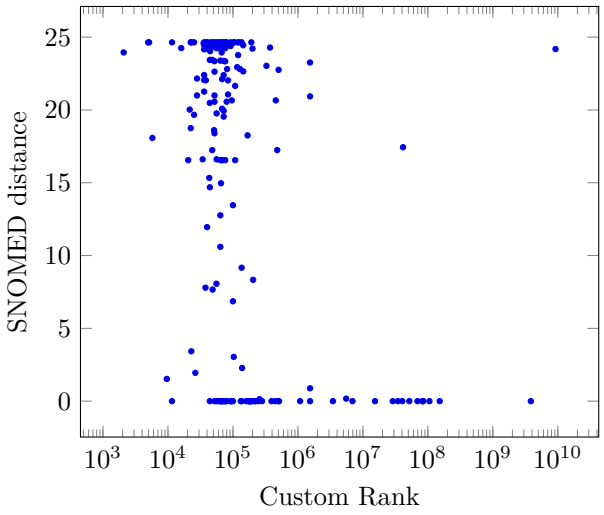

Figure 7: Result of document mapping the best FTS configuration

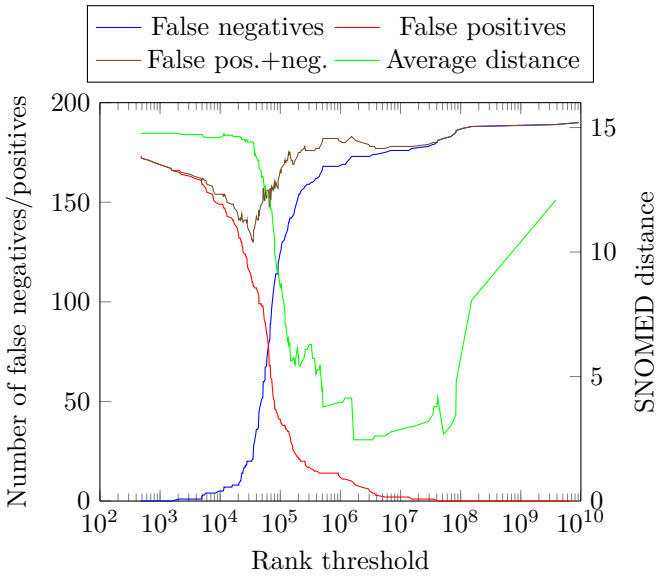

Figure 8: The false positives, false negatives, and average distance as a function of the custom rank

Name	Algorithm	Stoplist	Synonym	Direct hits	Avg. distance
Cust-stop-syn	Custom	Present	Present	17.3 %	14.768
Cust-stop	Custom	Present	Absent	17.3 %	15.090
Cust-syn	Custom	Absent	Present	13.4 %	15.765
Cust	Custom	Absent	Absent	14.4 %	15.400
FTR-stop-syn	FTR	Present	Present	5.8 %	18.909
FTR-stop	FTR	Present	Absent	6.5 %	18.691
FTR-syn	FTR	Absent	Present	3.9 %	18.566
FTR	FTR	Absent	Absent	3.7 %	18.489

Table 7: Results of all 8 full-text search configurations

The number of false positives, false negatives, and average distance as functions of the custom rank of Cust-stop-syn is seen in Figure 8. The average distance experiences quite a lot of noise for ranks higher than 10^8 as the available points decreases. For ranks greater than 10^5 the average distance is greatly reduced as the quality of the mapping increases. However, false positives are introduced at an equivalent rate, thus minimizing the amount of usable results. For the given verification set the sum of false positives and false negatives, also shown in Figure 8, indicates that the optimal threshold, giving the lowest sum, is $4{\cdot}10^4$.

5.4 Doc2Vec evaluation

In this section, we will evaluate and select the parameters for our doc2vec implementation. In Subsection 4.3, we presented several different decisions that might impact the performance of the model.

There are a number of parameters in the model which can be changed:

Data The data sets used for training from Table 6.

Preprocessing Whether we use the preprocessing methods: stemming and stoplist.

Size The number of dimensions in the vector space.

Window The length between words considered co-occuring.

Alpha Initial learning rate.

Min Alpha Target learning rate.

Algorithm Whether distributed memory is used or distributed bag of words is used.

Epochs Number of times the model is trained with the data.

We tuned these parameters by starting at the library defaults and independently changing each parameter to find a local optimum. The best parameters found can be seen in Table 9. In Table 8 we show that independently changing any of the parameters results in a model that is worse, or just as good as the one seen in Table 9.

A scatter plot of the mapping performed by the **best** model can be seen in Figure 9. We can see a weak negative correlation between the distance and the similarity reported by doc2vec. Thus, a higher reported similarity means that it is more likely that the mapping is close to the correct concept.

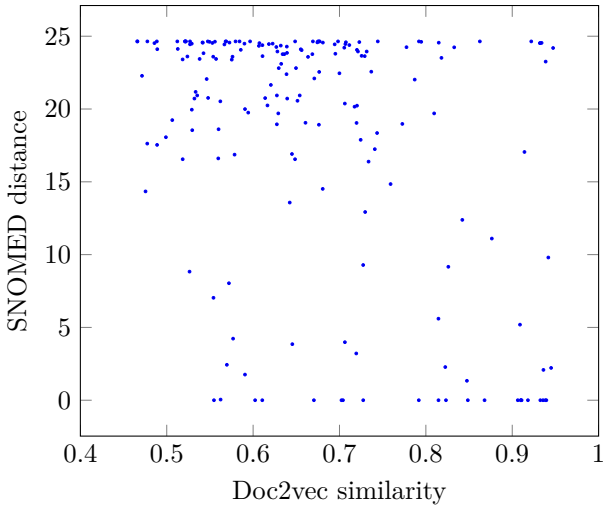

Figure 9: Result of document mapping for the "best" model

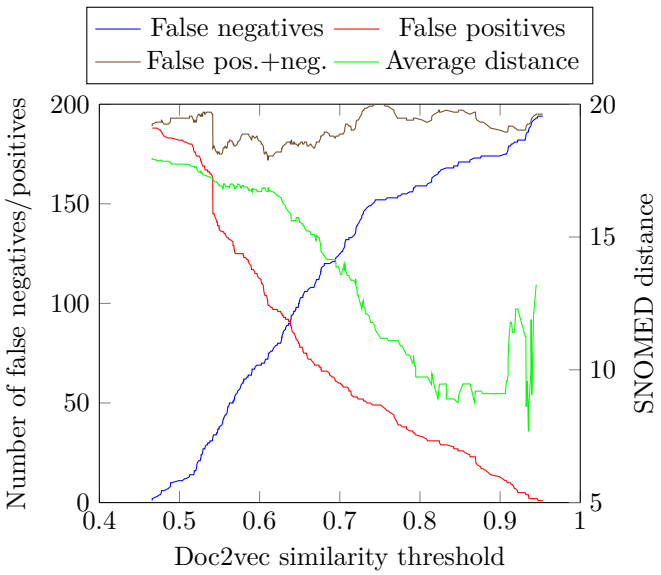

Figure 10: Effects of different thresholds for the "best" model

In Figure 10 we demonstrate the relationship between the number of false negatives, false positives, sum of false positives and false negatives, as well as the average distance as functions of the similarity threshold. We can see that the average distance descends as the threshold increases,

Parameter	Model name	Value	Direct hits	Avg. distance
Data	data-sno	SNO	0.0 %	21.844
	data-sno-sfi	SNO, EHR	5.5 %	18.522
	best	SNO, EHR, web	5.5 %	17.967
Preprocessing	token-none	-	4.7 %	18.600
	token-stop	Stop	4.2 %	19.195
	token-stem	Stem	4.7 %	18.469
	best	Stop, Stem	5.5 %	17.967
Size	size-50	50	4.7 %	18.714
	best	75	5.5 %	17.967
	size-100	100	5.2 %	18.054
Window	window-6	6	5.2 %	18.466
	best	8	5.5 %	17.967
	window-10	10	5.2 %	17.916
Alpha	alpha-015	0.015	5.5 %	18.119
	best	0.025	5.5 %	17.967
	alpha-035	0.035	4.5 %	18.408
Min alpha	minalpha-004	0.004	4.2 %	19.173
	best	0.005	5.5 %	17.967
	minalpha-007	0.007	6.0 %	18.124
Algorithm	**best**	DM	5.5 %	17.967
	dm-0	DBOW	1.3 %	21.632
Epochs	epochs-5	5	5.2 %	18.547
	best	10	5.5 %	17.967
	epochs-15	15	5.8 %	18.198

Table 8: Results of different parameters

Parameter	Value
Data	SNO, EHR, web
Preprocessing	Stopwords, stemming
Size	75
Window	8
Alpha	0.025
Min alpha	0.005
Algorithm	DM
Epochs	10

Table 9: Best model parameters

until after 0.9 where it increases again. This increase is likely caused by the noisier average distance as the number of points above the threshold decreases. The average distance above 0.9 is thus more susceptible to random variation. The sum of false positives and false negatives does not vary greatly for our verification set, but a threshold value of 0.61 gives the minimum.

6 Conclusion

The solutions presented in Section 4 have been evaluated in Section 5 and the results show that both of the presented solutions are better than the baseline methods mentioned in Subsection 5.2. The doc2vec method yielded an average distance of 8.9 at best with a threshold of 0.88 and FTS yielded an average distance of 2.5 at best with a threshold of $1.7 \cdot 10^6$. Comparing these results to the baseline with an average distance of 12.2 for the naive mapper, we can see that the average distance is better for both the doc2vec and FTS solutions. Another thing to consider when comparing these results, is the number of matching mappings, as the naive mapper will never map to the correct concept, where FTS and doc2vec managed to correctly map 17.3 % and 5.5 % of the concepts, respectively.

As discussed in Subsection 5.1.2 adjusting the threshold greatly affects the number of false positives and false negatives, as well as the average distance. As these attributes are not weighted equally in all situations, a threshold value can be chosen to best fit a given situation based on the graphs seen in Figure 8 and Figure 10.

The solution can likely be improved by implementing additional features. One of these, is the use of a thesaurus in the doc2vec method which could be beneficial to the correctness of the mapping. Future research could investigate other methods for mapping entries containing several concepts. In this paper the only method investigated is splitting the free-text into fragments. This might not be the best method, and it might also be possible to improve our implementation in this aspect. A key feature for an eventual production application would be to investigate the use of active learning, where the doc2vec solution could be improved iteratively by having an expert verifying or teaching the solution correct mappings.

7 Acknowledgments

We would like to thank Enversion A/S and the Central Denmark Region for their collaboration and support re-

garding the anonymized data used for our paper as well as presenting the project to us. We would also like to thank Simonas Šaltenis for excellent supervision throughout the project period.

.

References

[1] IHTSDO. *SNOMED CT Starter Guide*, version: 2014-02-22 edition, February 2014. URL http://ihtsdo.org/fileadmin/user_upload/doc/download/doc_StarterGuide_Current-en-US_INT_20140222.pdf.

[2] Microsoft Developer Network. *Full-Text Search*. Microsoft. URL https://msdn.microsoft.com/en-us/library/ms142571.aspx.

[3] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

[4] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

[5] Tua Holm, Anders Norr, Daniel Karlsson, and Mikael Nyström. Lessons learned from starting to implement snomed ct. In *SNOMED CT Implementation Showcase 2014-SNOMED CT: making health records make sense 30th & 31st October 2014, KIT (Royal Tropical Institute), Amsterdam, Netherlands*, 2014.

[6] Baolin Peng, Kaisheng Yao, Li Jing, and Kam-Fai Wong. Recurrent neural networks with external memory for spoken language understanding. In *Natural Language Processing and Chinese Computing*, pages 25–35. Springer, 2015.

[7] Holger Stenzhorn, Edson Jose Pacheco, Percy Nohama, and Stefan Schulz. Automatic mapping of clinical documentation to snomed ct. 2009. URL http://ebooks.iospress.nl/publication/12644.

[8] IHTSDO. Snomed ct - the global language of healthcare. URL http://www.ihtsdo.org/snomed-ct.

[9] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.

[10] Dan Jurafsky and Christopher Manning. Natural language processing. URL https://class.coursera.org/nlp/lecture.

[11] Microsoft TechNet. *How Search Query Results Are Ranked (Full-Text Search)*. Microsoft. URL https://technet.microsoft.com/en-us/library/ms142524(v=sql.105).aspx.

[12] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. Using of jaccard coefficient for keywords similarity. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, page 6, 2013. URL http://www.iaeng.org/publication/IMECS2013/IMECS2013_pp380-384.pdf.

[13] Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc, 2009.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[16] John Rupert Firth. *A Synopsis of Linguistic Theory, 1930-1955*. Blackwell, 1957.

[17] Vijay N Garla and Cynthia Brandt. Semantic similarity in the biomedical domain: an evaluation across knowledge sources. *BMC bioinformatics*, 13(1):261, 2012.

[18] Hassan Saif, Miriam Fernandez, Yulan He, and Harith Alani. On stopwords, filtering and data sparsity for sentiment analysis of twitter. 2014. URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/292_Paper.pdf.

[19] *Medicinske fagudtryk*. FOA - Fag og Arbejde, February 2005. URL http://abm.arbejdermuseet.dk/AbaFOAWeb/64896.pdf.

[20] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[21] Gordon Mohr. Gensim doc2vec & imdb sentiment dataset. URL https://github.com/piskvorky/gensim/blob/e7280954c1b2dfd99186ae64ae66e22005c5b22b/docs/notebooks/doc2vec-IMDB.ipynb.