Barriers and challenges of using medical coding systems
Surján, G.

Citation for published version (APA):
Surján, G. (2011). Barriers and challenges of using medical coding systems

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Chapter V. USING N-GRAM METHOD IN THE DECOMPOSITION OF COMPOUND MEDICAL DIAGNOSES

Abstract.

Objective: Our goal in this study was to find an easy to implement method to detect compound medical diagnoses in Hungarian medical language and decompose them into expressions referring to a single disease. Methods: A corpus of clinical diagnoses extracted from discharge reports (3079 expressions, each of them referring to only one disease) was represented in an n-gram tree (a series of n consecutive words). A matching algorithm was implemented in a software which is able to identify sensible n-grams existing both in test expressions and in the n-gram tree. A test sample of another 92 diagnoses was decomposed by two independent humans and by the software. The decompositions were compared to measure the recall and the precision of the method. Results: There was not full agreement between the decompositions of the humans, (which underlines the relevance of the problem) A consensus was arrived for all disagreements by a third opinion and open discussion. The resulting decomposition was used as a gold standard and compared to the decomposition produced by the computer. The recall was 82.6% the precision 37.2%. After correction of spelling errors in the test sample the recall increased to 88.6% while the precision slightly decreased to 36.7%. Conclusion: The proposed method seems to be useful in decomposition of compound diagnostic expressions and can improve quality of diagnostic coding of clinical cases. Other statistical methods (like vector space methods or neural networks) usually offer a ranked list of candidate codes either for single or compound expressions, and do not warn the user how many codes should be chosen. We propose our method especially in a situation where formal NLP techniques are not available, as it is the case with scarcely spoken languages like Hungarian.

1. Introduction

Providing assistance to the laborious and error prone work of human coding of clinical diagnoses by information systems is an old task in medical informatics. Since the pioneering work of Wingert [1] many useful methods were published, but practically none of them spread out from the local environment where it was born. We consider the logical process of coding as a multi-step abstraction process [2]. The input of this process is the clinical diagnosis. Clinicians comprise the essence of their knowledge about the patient into diagnoses. This is the original notion of the Greek word 'dia-gnosis'. Such diagnostic expressions sometimes refer to more than one disease. We call such expressions compound diagnoses. The frequency of compound diagnoses may depend on local traditions. In the environment studied by us (the Haynal University of Health Sciences) we observed compound diagnoses roughly in 20-30 % of all cases. Contrary to diagnosis the notion of
Chapter V – N-gram method

disease refers to an abstract category to which certain cases belong. This difference explains why the number of diagnoses and diseases of a patient do not necessarily agree. During the indexing process there is a temptation to assign to one diagnostic expression exactly one code regardless of the number of diseases the diagnosis refers to. This may lead to errors in hospital morbidity statistics and to inappropriate financing. (Hungarian hospitals are financed on the basis of an adapted DRG system). If 20-30% of the diagnoses are compound and at least each third compound diagnosis is miscoded, then the statistical error is about 7-10%. Considering that the usual error rate in ICD coding is about 25-30%, it is a significant source of errors.

In this study we seek for an easy to implement solution, which is able to recognise composite diagnoses and identify their components, the ‘atomic’ disease concepts. The proposed n-gram based method is statistical and language independent, but not fully automatic. It is not our goal to exclude all human activity from the coding process, but to reduce coding errors by effective assistance of the work of human coders.

2. Method

In the following:

1. The term ‘Word’ denotes an ‘elementary unit of information’, in other words it is a symbol, which carries meaning and its internal structure is irrelevant in the given approach.
2. Any complete sequence of Words semantically related to each other will be called ‘Sentence’.
3. In this paper by n-gram we mean \( n \) consecutive Words of a particular Sentence \[3\], not a string consisting of a consecutive letters. (The term n-gram is also used in the literature sometimes for a string consisting of a consecutive letters \[4\]. Note the difference!)

We capitalise ‘Sentence’ and ‘Word’ to discriminate these concepts from word and sentence as grammatical units of language. For instance a code and the related diagnosis together form a Sentence, which grammatically may be a noun phrase. The code and the words of the diagnostic expressions together form the Words of this Sentence.

Our corpus-based method needs a sample of error free, clinical diagnoses. An ideal sample should contain all possible diagnoses of a certain medical domain expressed by the words and phrases used in the daily clinical practice. E.g. lists taken from classification systems like ICD can not be used because their language is different from the clinical one. Therefore we decided to collect a sample of clinical diagnosis expressions from a large number of patient discharge reports. We call it reference base (\(R\)). The aim is to have Sentences containing only sensible, properly spelled diagnoses, all expressing exactly one disease. The last criterion is vital for being able to decompose compound expressions. As the method is also planned to be used as a computer assisted coding tool, we added the appropriate ICD code of the diseases to the Sentences. The use of codes will be explained later.
2.1. The n-gram tree

Once we have such a reference base, all n-grams for each Sentence of the sample are generated (from length of one up to the length of the particular Sentence). In other words, we store all the continuous fragments of each Sentence. The typical use of n-grams is to disambiguate ambiguous words and spelling error correction [3]. We use them to describe the valid sequences of words in clinical diagnoses. Since the reference base does not contain several hundred thousand diagnoses and the diagnostic expressions are typically short (4 - 5 words) we are not forced to use a fixed n (typically 3 in the literature) but can build n-gram statistics using a dynamic n: the length of the particular expression.

A special tag "_FULL_" is added to the n-gram representing the whole Sentence, which enables to identify complete Sentences. The included code of the given diagnosis expression is marked by pairs of $ signs, and this marked code is added to each of its n-grams. The co-occurrence frequency of the code and the n-gram is also stored. The following example illustrates the decomposition:

Coded expression: $E10.9$ diabetes mellitus type-I

N-gram decompositions:

- $E10.9$ Diabetes
- $E10.9$ mellitus
- $E10.9$ type-I
- $E10.9$ diabetes mellitus
- $E10.9$ mellitus type-I
- $E10.9$ diabetes mellitus type-I _FULL_

By decomposing all Sentences of the reference set we get a reference n-gram set. For practical reasons - such as saving memory and quick searching - the n-gram set is stored in a tree structure. This is illustrated in Figure V-1. The following Sentences are represented in the illustration:

- $E14.9$ Diabetes mellitus
- $E10.9$ diabetes Mellitus type-I
- $E11.9$ diabetes Mellitus type-II
- $E23.2$ diabetes insipidus

Since codes always appear together on the same level as the _FULL_ marks in Figure V-1 either the code or the FULL mark is shown. (Codes and FULL marks are tags, and in the matching process they are not considered as a part of the tree.) Starting from the root of the tree each node of the tree describes one n-gram. The nodes are arranged into layers. In the given example we have 4 layers, separated by dashed lines and numbered at the right side of the Figure. The importance of this arrangement will be explained later. The dotted line illustrates the n-gram "mellitus type-II". The number in the boxes defines the number of the occurrences of the corresponding n-gram. E.g. "mellitus type-I" occurs once, while "Diabetes mellitus" occurs three times in the sample. (Look at the "mellitus" typed in bold in the second layer.)
Figure V-1 The n-gram tree

2.2 The initial matching process

Once an n-gram tree is created from a proper reference set, we can take a new diagnostic expression as a test Sentence (T). The goal is to find the longest matching and meaningful n-grams so that all words of the test Sentence shall be covered. The matching process starts with the last word of the test Sentence. Let say, the test Sentence consists of k words. At first we try to find the last word of T in the k-th layer of the tree. If it is successful then we go upwards in the tree to the k-1-th layer, and check if the next node in the tree is present in the next (or surrounding) 2 words in the test Sentence. If it is so, we repeat this process until we arrive at the root. All matched words are marked. A matching process is complete, if all the words of the test Sentence are marked as matched. If the match is not complete, the whole matching procedure will be started again with the last not matched word, in the p-th layer, where p is the number of not matched words. Of course it can happen that the last word of the test Sentence cannot be found in the k-th layer of the tree. In this case we take the last but one word of the test Sentence and start the matching again from the k-th layer. If none of the words of the test Sentence can be found in the k-th layer, we move to the k-1-th layer and start the matching from here with the last Word of the Sentence.

It is possible that some words in the test Sentence were not present in the reference base and therefore were not present in the n-gram tree. Such ‘unknown’ words are simply skipped in the matching process. A word can be unknown because of spelling error. Really unknown words may occur, if the reference sample does not cover the given medical domain, or the test Sentence contains concepts outside of that domain. If an unknown word alone represents a disease, the identification naturally fails. If the unknown word is only a part of a diagnosis,
there is still some chance to recognise the diagnosis by the other words of the expression. When unknown words occur frequently the reference base should be extended.

At the end of the whole matching process we retrieve a set of n-grams. In the simplest case the match is complete and we get just one n-gram (complete single match). It is also possible that the match is complete, but the words of the test Sentences are covered by more than one n-gram (complete multiple match). Another issue is whether the retrieved n-grams are "FULL" n-grams (FULL match) or not. So the result of the matching process can be as follows:

a) One FULL n-gram, complete match. We can infer that the diagnosis expresses one single disease.
b) More than one FULL n-gram, complete match. We can infer that the diagnosis expresses more than one disease.
c) One or more n-grams, complete match, at least one of the n-grams is not FULL. Not FULL n-grams must be searched further for completing (see the next section)
d) Incomplete match: at least one word was unknown.

Remember that all single words form an n-gram itself. So if the test Sentence would consist of a random selection of \( k \) completely unrelated words, then we would get \( k \) n-grams, each consisting of one word, and this still would be a complete match.

In principle by completing the initial matching process we may achieve our main goal: recognising compound diagnoses. Once a diagnosis comprises two or more diseases, it is very likely that the matching process retrieves more than one n-gram (multiple match), since each disease will match a different n-gram. For two reasons it is worthwhile to search for more n-grams by extension of the matching process. First, if the initial match was FULL and complete, it still may happen that the found concepts have subtypes to be considered for coding. (E.g. the test Sentence was "external otitis" but "malignant external otitis" also may be the most appropriate category. Second, if at least one n-gram was not FULL the extension process described in the next section may reveal the relevant FULL n-gram.

2.3. The extension process

2.3.1. Downward extensions

The result of the initial matching process is one or more n-grams. They can be either FULL or not. In both cases it may happen that such n-grams still have sub-branches in the tree. For instance, the test Sentence "Diabetes" retrieves a non FULL n-gram, while "Diabetes mellitus" retrieves the FULL n-gram. In the first case the test Sentence is only one word: "Diabetes". (This is a colloquial expression for diabetes mellitus, but in principle the expression is ambiguous, since it may refer to 'diabetes mellitus' and 'diabetes insipidus' [insipid diabetes] as well, which are rather different entities. ) In this case the matching process will find the word
'diabetes' in the first layer, and this is immediately a complete - but not FULL - match. The downward extension process will retrieve the following strings:

- Diabetes + mellitus +_FULL_
- Diabetes + mellitus type-I +_FULL_
- Diabetes + mellitus type-II +_FULL_
- Diabetes + insipidus +_FULL_

The test Sentence "Diabetes mellitus" retrieves the FULL n-gram

Diabetes+mellitus+$E14.9$+FULL"

But even this FULL n-gram has sub-branches starting from the node

 Diabetes + mellitus + type-I +$E10.9$+_FULL_ and Diabetes + mellitus type-II +_FULL + $E11.9$

The downward extension retrieves all FULL tagged sub branches.

Diabetes mellitus is a correct but not fully specified diagnosis. If the possible extensions are offered to the human coder, then he can realise that there are some more specific diagnoses here, and if the necessary information is available, it is possible to choose the most specific one. (As it is declared in the ICD coding rules.)

It is also possible that the "FULL" match was impossible due to spelling errors. Let say the test Sentence is 'Diabetes me

litus'. In this case the misspelled 'melitus' will be skipped in the initial matching process as an unknown word, and the n-gram 'Diabetes' will be retrieved. In such cases the unknown words can be compared to the words occurring in the extensions. Since 'melitus' is more similar to 'mellitus' than to any other words in the extensions, this is the most likely correction of the spelling error. (This similarity can be measured by the length of common sub-strings in identical order.) Once the user accepts such a proposed correction, the test can be repeated with the corrected test Sentence.

2.3.2. Upward extensions

The above-described downward extension does not help, if the first word is missing from an expression by mistake.

Suppose the word diabetes is missing from the expression diabetes insipidus. In this case the matching algorithm will find the n-gram 'insipidus' from the first layer of the tree, but this n-gram is not "FULL". But there is no way downwards in the tree. To find the missing word, the algorithm goes to the k+1-th layer (this is the second layer in our case) and tries to go back to the root. Contrary to the normal matching process, in upward extension the algorithm will not fail at the first unsuccessful match. The test sentence will be extended with the first unmatchted word.

The upward extension process can be performed recursively, and each step is followed by a downward extension.

At the end of the whole extension process the resulting list of alternative n-grams can be too long for humans to review. In this case it is possible to rank them by their frequency in the reference base. In our example it is likely that the diagnosis correction
contains much more "diabetes mellitus" then "diabetes insipidus", since the former is more common.

2.4. The word order problem

In principle, the matching process is order independent. Especially in case of abbreviations the order of words may change without any change of the meaning. Both 'Chr. Tons' and 'Tons. Chr.' refer to chronic tonsillitis. Suppose, the reference set contains only one of them - let say 'Tons. chr'. - and the test Sentence is the other one. The matching process will start at level 2 with the last word of the test Sentence ('Tons.' in our case). But this word will not be found at level 2, so as it was described, the matching process will continue with the next word of the test Sentence ('chr.') and this one will be matched. Going upwards in the tree the 'Tons.' will be found at level 1 and this is given in the test Sentence, so the root is achieved. Now the match is complete.

In certain cases the order independent matching may lead to confusion. Let us take the following Sentence as an example:

'Insulin dependent diabetes, non Hodgkin lymphoma'

In this case the matching process will start at level 6, but the first match will be found only in level 4 with the word 'diabetes'. If the matching would be totally order independent, we would go back from this node to the root and we would retrieve the following n-gram:

'non insulin dependent diabetes'.

Then the matching process would be continued with the rest of the words, and 'Hodgkin lymphoma' would be retrieved as well.

To avoid this type of confusion, the matching is restricted to the surrounding two words of the last matched word. So when the word 'insulin' is matched at level 2, and the next word in the tree is 'non', the matching process stops, because although the word 'non' is present in the test Sentence the distance is too large. Therefore the whole process will be started again from level 3, where the word 'diabetes' is also present, and successfully will go back to the root, retrieving the n-gram 'Insulin dependent diabetes'. The rest of the words will lead correctly to the n-gram 'non Hodgkin lymphoma'.

Here the question arises why the surrounding two words are used and not three or four etc. This is an arbitrary choice of course. We tested the method with various settings and got the best results with this one. But we have to realize, that the size of the environment is not critical. Sometimes it may lead to failure, but is compensated by the extension process. Let us take the following example:

'sinusitis, ethmoiditis et tonsillitis acuta'
In this example the term ‘acuta’ refers to ‘sinusitis’ and to ‘ethmoiditis’ as well, but the matching process will stop at the single word ‘ethmoiditis’ since the distance between ‘acute’ and ‘ethmoiditis’ is too long. However since the match will be not "FULL", the extension process will produce the required expression ‘acute ethmoiditis’ as one of the possible completions of the expression.

2.5. The reference base

For the experiments performed to test the method, a reference base was created. It contained 3079 diagnoses collected from discharge reports from the various clinical departments of the Haynal University. These diagnoses were carefully coded into ICD10 [6] by human experts. This set of diagnoses was used by the authors in various experiments [7, 8]. Errors found during these experiments were corrected manually. The generation of the n-gram tree from the 3079 diagnoses took a few seconds on a Pentium III 500MHz machine. The tree contains 14995 nodes, in the first layer there are 2940 nodes (it equals the number of the different words in the reference base), the maximal depth of the tree is 17.

The decomposition of a Sentence took only a couple of milliseconds, so the computing time is suitable for daily practice.

2.6. Software solution

The method has been implemented by the authors in Visual C++, no commercially available database manager software has been used (for the sake of speed). The program responsible for generating the tree of the used reference base is able to process arbitrarily large sets of Sentences, the tree is stored in a binary file.

The output of the system is a ‘tab’ delimited file importable to Microsoft Excel for further analysis.

The software has also a user interface to analyse Sentences one by one on the screen.

3. Experiment

3.1. Setting up the gold standard and the test

To test the method we need a gold standard: a set of clinical diagnoses containing both simple and compound diagnoses, which is decomposed manually in a controlled way.

We took a collection of 92 randomly selected different diagnoses from clinical discharge reports. The reports used for the reference base and the gold standard were collected from the same departments of the Haynal University of Health Sciences. Two independent experienced encoders were asked to decompose the diagnostic expressions, when they considered them as compound, into terms expressing a single disease [5].

There was not full agreement between the two human encoders concerning the question, which diagnoses were compound, and which were simple. Out of the 25 diagnoses considered as compound by at least one encoder their judgements were different in seven cases. In three cases encoder B considered the expression as a
simple disease, while encoder A considered them as compound. In further four cases the decomposition was not identical. This shows that the unawareness of compound diagnoses is the cause of coding errors in a significant number of cases. The results of the two decompositions were reviewed by the authors and we came to a consensus with the two coders. This final decomposition is used as gold standard for the test. Out of the 92 diagnoses 67 were considered as simple, 25 were considered as compound. The original 92 diagnoses were decomposed into 132 simple diagnoses.

The same original 92 diagnoses were used as test sample. So the decomposition made by the computer and the gold standard decomposition could be compared.

3.2 Evaluation

To evaluate the decomposition made by the computer two measures were used, called precision and recall. We used them in the following sense:

\[
recall = \frac{\sum b_i}{g} \quad (1)
\]

\[
precision = \frac{\sum b_i}{c} \quad (2)
\]

where \( b_i \) is the number of expressions found both in the computer and in the gold standard decomposition of the \( i \)-th test sentence, \( g \) is the total number of expressions in the gold standard and \( c \) is the total number of expressions found by the computer decomposition.

The recall signifies how much we can rely on the system, while precision says how easy it is to find the appropriate expressions in the retrieved list.

4. Results

The system decomposed the 92 diagnostic expressions into 293 expressions. In total 109 expressions were also present in the gold standard. So the recall was 82.6 %, and the precision was 37.2 %. (As it was mentioned, the gold standard consists of 132 simple diagnostic expressions.)

In the case of 21 diagnoses in the test sample some spelling errors were found. Since the approach helps to correct spelling errors, they were corrected. After the correction the system decomposed the 92 diagnoses to 319 expressions. From these expressions 117 were found in the gold standard (valid expressions). The recall is increased to 88.6 % and the precision is decreased to 36.7 %. All together, 15 expressions from the gold standard could not be matched by the approach. In 14 cases the reason was that the expression contained essential words unknown for the reference base. Out of these expressions ten referred to a medical procedure,
Chapter V – N-gram method

not a disease. Physicians sometimes like to refer to history of a certain medical procedure in clinical diagnoses. E.g. "Status post appendectomiam" (means "Status after appendectomy", Latin). Such expressions were excluded from the reference base during previous work.

Only 1 expression was missing due to an error of the matching algorithm. In this case the algorithm found a word in the surrounding which semantically did not belong to the expression, but the matching algorithm led to a sensible other expression.

From the 117 valid expressions 62 were found by the direct matching algorithm, 53 was found in the extended n-grams.

202 expressions generated by the system were not present in the gold standard (false positives).

5. Discussion and conclusion

Methods for document retrieval and computer assisted coding of diagnoses are very similar. Wiesman et al. gives a good overview of these methods [9]. Some of them like vector-space methods are able in principle to identify compound diagnoses and find codes for all included diseases. However, if a given implementation of these methods is not designed for this purpose, you can never be sure whether the multiple codes come from mistakes or from the fact, that the diagnosis is compound. The authors do not know any work in the literature directly addressed to the problem of compound diagnoses.

Formal solutions based on natural language understanding and formal conceptual representation may be more effective, but very hard to implement in the Hungarian medical language. These facts justify our approach.

In all corpus-based statistical approaches two questions arise. One is the potential benefit of ‘canonisation’ of terms, i.e. unification of orthographical and lexical variations, etc. The other question is that the capability of the approach strongly depends on the size and quality of the corpus.

Concerning the first problem we performed an experiment, where all words were stemmed, synonyms and language variations were replaced by a preferred term, and references to laterality were standardised. (Discrimination of left and right is irrelevant for a classification. Uni- and bilateral conditions were discriminated only.) Surprisingly the results after this canonisation were practically the same. The recall was the same but the precision became slightly lower.

Our results clearly show that the recall of the method could be even higher if the reference base was more complete. On the other hand the precision presumably would decrease, because of the increasing number of possible combinations in a larger n-gram tree. The low precision does not weaken seriously the usability of the method, since we do not intend to use it as an automatic process. Our philosophy is always to assist the human encoders, who can easily select the appropriate expressions from a limited set of offered possible extensions. However the review of a long list is bothering to the user therefore the precision should be augmented and the results should be ranked according to their possibility of being valid.
Further research is necessary to find ways to increase the precision of the approach. The presented method was easy to implement and did not require expensive knowledge engineering or language processing work. However, it is necessary to create a reference base, consisting of manually coded diagnoses. Since it is possible to increase incrementally the size of the reference base, the method is scalable: more and more medical domains can be included gradually.

The authors had some experience in computer assisted coding with vector space [7, 10] and neural network (ANN) [8] methods. In case of compound diagnoses those methods may suggest codes for all included diagnoses but cannot warn the user to choose multiple codes. Vector space and ANN methods usually offer a ranked list of codes, and do not inform the user whether one or more codes should be selected by the user.

Acknowledgements

Our special thanks to Tadeusz Dobrowiecki and Arie Hasman for their kind help and useful remarks.

References