Extending the Query Language of a Data Warehouse for Patient Recruitment

Georg DIETRICHa,[[1]](#footnote-1) Maximilian ERTLc, Georg FETTEa,b, Mathias KASPARb, Jonathan KREBSa, Daniel MACKENRODTd, Stefan STÖRKb and Frank PUPPEa

a *Chair of Computer Science VI, Würzburg University, Germany*

b *Comprehensive Heart Failure Center, University Hospital Würzburg, Germany*

*c Service Center Medical Informatics, University Hospital Würzburg, Germany*

d *Institute for Clinical Epidemiology, Würzburg University, Germany*

**Abstract.** Patient recruitment for clinical trials is a laborious task, as many texts have to be screened. Usually, this work is done manually and takes a lot of time. We have developed a system that automates the screening process. Besides standard keyword queries, the query language supports extraction of numbers, time-spans and negations. In a feasibility study for patient recruitment from a stroke unit with 40 patients, we achieved encouraging extraction rates above 95% for numbers and negations and ca. 86% for time spans.

**Keywords.** Information Extraction, Data Warehouse, clinical trials, text queries.

# Introduction

A comparison of electronic health record (EHR) systems showed that patient recruitment is not a standard feature and that clinical research has not been in the focus of the EHR vendors [1]. Clinical Data Warehouses (CDW) like I2B2 [2] or PaDaWaN [3] are very suitable for patient recruitment for clinical trials with inclusion and exclusion criteria.

Steps towards an automatic identification of patients for clinical trials by parsing the natural language description of the criteria has been designed [4] and attempted [5]. Both approaches queried existing structured data only.

But a data warehouse also consists of unstructured data like textual reports and discharge letters, that - with the exception of search engine technologies [6] - is hardly usable for this task without preprocessing. There are many attempts to extract structured information from unstructured documents with various approaches (e.g. [7], [8], [9] for German texts). However, such information extraction approaches must be done when the data is loaded into the data warehouse and require a lot of engineering work for preprocessing. Therefore, a query option where rules for search criteria can be stated ad hoc even on unstructured data would be an attractive feature for a data warehouse.

However, this topic has not been studied well for CDWs. The text query features of I2B2 are not published officially, but the documentation shows, that text data can be queried with the “like” operator of SQL[[2]](#footnote-2). That is similar expressive as wildcard queries. This search is a filter, it does not extract data. In addition, it is much faster to query an indexed document than to use the SQL like operator processing the entire text.

This paper describes a query extension for the CDW PaDaWaN of the University Hospital Würzburg and a feasibility study in which patient recruitment will be done with complex ad-hoc queries for patients from a stroke unit. The input is a query containing the inclusion and exclusion criteria and the output is a list of suitable patients. It is intended, that an alert is generated, when a new potential participant has been found for the clinical trial. Challenging criteria for the recruitment include a rule condition on the duration from onset of the stroke till admission at the hospital and some exclusion criteria like atrial fibrillation.

# Methods

The automatic patient recruitment system is a pipeline consisting of a query tool for the CDW that produces a spreadsheet with all relevant information which is then logically processed by an interpretation layer. The outcome are patients which are a candidates for recruitment (Figure 1).

pipeline.png

**Figure 1.** Pipeline of the recruitment system.

PaDaWaN uses as storage engine the index library Lucene[[3]](#footnote-3) containing both structured and unstructured data. That’s a big advantage for dealing with textual queries. During the index process, every piece of information runs through a data type specific pipeline for preprocessing before it is added to the index. The pipeline for text data contains basic NLP operations like stemming, stop word removal and ignoring case sensitivity.

The most important development took place in three different stations in the query progress. First we added a useful **preprocessing** function, second we added several powerful **query features** and third we modified the **result presentation** to make these improvements usable for further processing.

The additional preprocessing function is used to identify all negations and pseudo negations in a text by using a modified version of the negex algorithm [10]. The text is segmented in noun phrases which are classified whether they contain a negation or not.

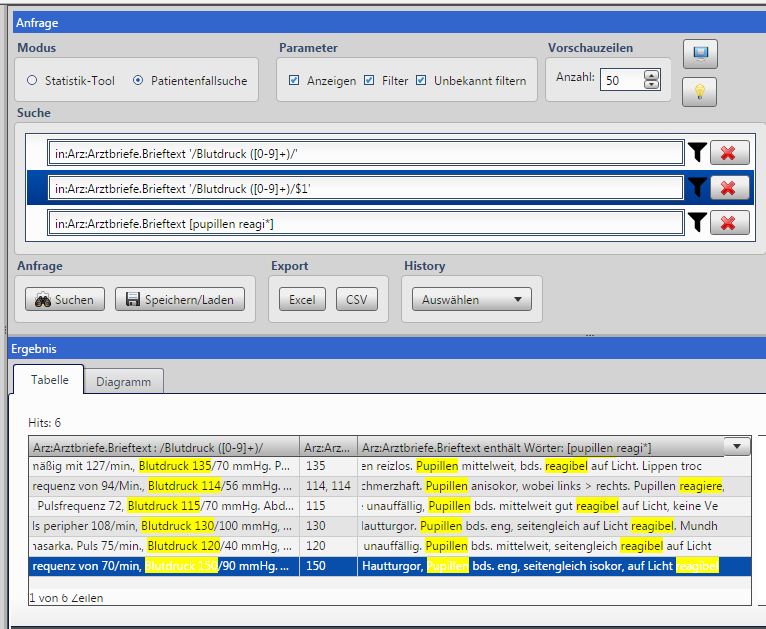
Furthermore, discharge letters get segmented into sections like history, physical examination, lab data, etc. that are queryable separately afterwards.

The additional query features include basic, but strong functions like boolean retrieval, wildcards and phrase query. Boolean retrieval returns texts that contain some given words and can be combined using the logical operators *and, or, not.* Wildcards can be used at any position of a token to represent any characters. A phrase query matches a sequence of words like e.g. "Vorhofflimmern" (atrial fibrillation):

*vhf OR vorhofflimm\* OR vfli OR vofli OR fvhli*

A more advanced feature is a context specific query providing control over the order and proximity of the given terms. The input are two or more words that must be in a context to each other in the queried text, like e.g.:

*[Flimmern Vorhof]*



**Figure 2.** Example query in the PaDaWaN web app: First column shows a regex query which is displayed with context in the result snippet, second shows the extracted value of the regex query. The last column shows a context sensitive query with a wildcard.

Another powerful feature is the query with regular expressions (regex). The user can define a regex using the standard syntax including predefined character classes, quantifiers, alternatives and grouping. For subsequent processing the format of the output can be defined, too: the whole expression with context, just the expression or even a single group, e.g. a number, e.g.:

*/parameter ([0-9]+)/$1*

The expression is defined between the slashes. In this example, it must start with the string "parameter". The brackets define a group, here *[0-9]+* which represents a number. *$1* refers to the first group in the regex, the number. So the output of this regex is just the number that can be further processed by rules (see interpretation of results in Figure 1).

For dealing with the extraction of dates, many such expressions are necessary. To increase the usability a subquery mechanism allows splitting big queries into smaller groups. They can be stored separately and then referenced from other queries. That allows an incremental creation of complex queries.

The result presentation visualizes the results for the diverse query features. The hits of the given query terms are highlighted in matching documents from which small snippets are generated similar to Google. Optionally, the underlying entire text (i.e. the report) can be opened and the hit is highlighted in that as well. Each column of the result can also be exported in a CSV or Excel format (see Figure 2).

The query language is kept simple so that the physicians can use the tool on their own. They work together with technical experts for complex queries.

The interpretation layer in Figure 1 is a simple rule framework which transforms words like *noon* to *12 pm* or computes time differences between two timestamp.

# Results

We evaluated the performance on 40 patient cases in the context of a stroke study recruitment task with emphasis on four different query features: standard keyword search, search with negations (atrial fibrillation negated), extraction of a numerical value (used by rules in the interpretation layer) and extraction of two timestamps of a day (used by the interpretation layer to compute a time span and compare it with a threshold). For this study we used no structured data, all information was extracted from text documents. Not all desired data was given within each patient case document, therefore we give the number of relevant documents in addition in Figure 3.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Documents | Correct extractions | % |
| Standard keyword search | 40 | 40 | 100 |
| Keyword search with negations | 40 | 39 | 97,5 |
| Extraction of numerical values | 39 | 38 | 97,4 |
| Extraction of the time of a day | 30 | 26 | 86,6 |

Figure 3. Evaluation results for four different query types within the feasibility study for patient recruitment

# Discussion

Our solution to deal with negations in texts was to restrict the scope of the query to just two sections instead of the entire discharge letter. Otherwise, more errors would have been made than 1 in 40 documents. However, we used knowledge about the particular use case. Another way facing this problem is to improve the negation identification in general at preprocessing time, but this is still very challenging [11, 12, 13, 14].

The biggest challenge in the feasibility study was the computation of the duration of symptoms. To achieve that two dates have to be parsed and subtracted. Errors were made by mapping words like *“morning”* or “*since awake”* to the specific date and time, because some words were not covered by the rules and the resolution of other dates was incorrect. For this use case, working with time intervals would be simpler and sufficient, because only the classification of the duration of two dates matters. For instance, the duration between since “*awake”* and “*noon”* could be categorised into “less than 10 hours”.

# Conclusion

This paper presents a lightweight but powerful instant information extraction method approach for recruiting patients for a clinical study. It allows to gain information from unstructured texts. The engineering process takes minutes instead of weeks or month, when the data is preprocessed by an information extraction approach.

Further development will be done to add features and to expand the expressive power. One specific goal of future work is the integration of the interpretation layer into the query tool, which is currently a post-processing step. Then the system would e.g. be able to perform interval arithmetic or complex calculations like BMI = weight / height².

# Conflict of Interest

The authors have no conflict of interests.

References

1. Schreiweis B, Trinczek B, Köpcke F, Leusch T, Majeed R, Wenk J, Bergh B, Ohmann C, Röhrig R, Dugas M, Prokosch H. Comparison of Electronic Health Record SystemFunctionalities to support the patient recruitmentprocess in clinical trials. International Journal of Medical Informatics, 83(11), 860-868, 2014
2. Murphy S, Weber G, Mendis M, Gainer V, Chueh HC, Churchill S, Kohane I. Serving the Enterprise and beyond with Informatics for Integrating Biology and the Bedside (i2b2), J Am Med Inform Assoc. 17(2) (2010) 124-130.
3. Dietrich, G., Fell, F., Fette, G., Krebs, J., Ertl, M., Kaspar, M., Störk, S. Puppe, F.: Web-PaDaWaN: Eine Web-basierte Benutzeroberfläche für ein klinisches Data Warehouse (2016). HEC 2016, Joint Conference of GMDS, DGEpi, IEA-EEF, EFMI, DocAbstr. 421, 2016.
4. Majeed R, Röhrig R.Identifying patients for clinical trials using fuzzy ternary logic expressions on HL7 messages. Stud Health Technol Inform, 169, 170-174, 2011
5. Lonsdale DW, Tustison C, Parker CG, Embley DW. Assessing clinical trial eligibility with logic expression queries. Data & Knowledge Engineering; 66(1): 3–17, 2008
6. Dietrich, G.; Fette, G.; Puppe, F.: A Comparison of Search Engine Technologies for a Clinical Data Warehouse. in CEUR Workshop Proceedings, T. Seidl, Hassani, M., Beecks, C. (eds.) (2014). (Vol. 1226) 235-242.
7. Starlinger J, Kittner M, Blankenstein O, Leser U. How to Improve Information Extraction from German Medical Notes, it - Information Technology 58 (10/2016).
8. Krieger H.-U., Spurk C., Uszkoreit H., Xu F., Zhang Y., Mueller F., and Tolxdorff T. Information extraction from german patient records via hybrid parsing and relation extraction strategies. In LREC, pages 2043-2048, 2014.
9. Toepfer M., Corovic H., Fette G., Kluegl P., Stoerk S., and Puppe F. Fine-grained information extraction from german transthoracic echocardiography reports. BMC medical informatics and decision making, 15(1):1, 2015.
10. Chapman W, Bridewell W, Hanbury P, Cooper GF and Buchanan BG, A simple algorithm for identifying negated findings and diseases in discharge summaries. Journal of Biomedical Informatics. Volume 34, Issue 5, October 2001, Pages 301-310
11. Wu S, Miller T, Masanz J, Coarr M, Halgrim S, Carrell D, Clark C. Negation’s Not Solved: Generalizability Versus Optimizability in Clinical Natural Language Processing. 2014. PLOS ONE 9(11)
12. Mutalik P, Deshpande A, Nadkarni PM. Use of general-purpose negation detection to augment concept indexing of medical documents: a quantitative study using the UMLS. J Am Med Inform Assoc. 2001 Nov-Dec;8(6):598-609.
13. Huang Y, Lowe HJ. A novel hybrid approach to automated negation detection in clinical radiology reports. J Am Med Inform Assoc. 2007 May-Jun;14(3):304-11. Epub 2007 Feb 28.
14. Sohn S, Wu S, Chute CG.; Dependency Parser-based Negation Detection in Clinical Narratives. AMIA Jt Summits Transl Sci Proc. 2012; 2012: 1–8.

1. Corresponding Author, Georg Dietrich, Chair of Computer Science VI, Würzburg University, Am Hubland, 97074 Würzburg, Germany; E-mail: georg.dietrich@uni-wuerzburg.de. [↑](#footnote-ref-1)
2. http://community.i2b2.org/wiki/display/DevForum/Text+search+in+i2b2 [↑](#footnote-ref-2)
3. https://lucene.apache.org/core/ [↑](#footnote-ref-3)