Human-in-the-loop Language-agnostic Extraction of Medication Data from Highly Unstructured Electronic Health Records

Frank Ruis*, Shreyasi Pathak*, Jeroen Geerdink†, Johannes H. Hegeman†, Christin Seifert*, Maurice van Keulen*
* *Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente*, Enschede, The Netherlands
† *Hospital Group Twente*, Hengelo, The Netherlands
Email: <s.pathak,c.seifert,m.vankeulen>@utwente.nl,
<j.geerdink,h.hegeman>@zgt.nl
f.a.ruis@student.utwente.nl

***Abstract*—Electronic health records contain important information written in free-form text. They are often highly unstructured and ungrammatical and contain misspellings and abbreviations, making it difficult to apply traditional natural language processing techniques. Annotated data is hard to come by due to restricted access, and supervised models often don't generalize well to other datasets. We propose a language-agnostic human-in-the-loop approach for extracting medication names from a large set of highly unstructured electronic health records, where we reach almost 97% recall on our test set after the second iteration while maintaining 100% precision. Starting with a bootstrap lexicon we perform a context based dictionary expansion curated by a human reviewer. The method can handle ambiguous lexicon entries and efficiently find fuzzy matches without producing false positives. The human review step ensures a high precision, which is especially important in healthcare, and is not subject to disagreements with annotations from an external source. The code is available online [1].**

***Index Terms*—dictionary expansion, context matching, medication extraction, human in the loop**

I. Introduction

Electronic Health Records (EHRs) contain a wealth of information on symptoms, diagnoses, treatments, test results, images, interpretations, and outcomes. Clinicians, however, primarily communicate with each other through letters and reports. So although EHRs do maintain quite some information in structured fields, much more information is actually only available in the form of unstructured text [14].

In this paper, we focus on medication information: obtaining an accurate history and current use of a patient's medication in a structured form. Even though limited information is typically available in structured form, it is not always reliable nor complete enough due to loss of information when patients move between institutions, limited record-keeping, and over-the-counter drugs. In an operational setting, a clinician would therefore also scan for appropriate sections in certain reports to get a reliable and more complete overview. For research and other analytical purposes, where reliable and complete data on thousands of patients needs to be gathered, this reliance on human reading and interpretation is a severe obstacle which leads to either researchers spending years to gather a minimal amount of data, or to abandonment of the effort and not using the available unstructured data at all.

There exist approaches for extracting medical information from EHRs, in part thanks to the i2b2 [10] and later the n2c2 [5] challenges. These range from rule-based [15] to supervised machine learning [12] and ensemble methods [6].

Besides the aspect of natural language, a significant challenge in EHRs is their quality. They are often written hastily while the patient is answering questions, resulting in misspellings, abbreviations, ungrammatical sentences, and inconsistent use of punctuation and line breaks. In other cases, dictation is used resulting in other kinds of errors and imperfections. Furthermore, due to privacy considerations, richly annotated datasets are hard to come by. And the few that are available are often not usable as they are mostly in the English language and, in general, supervised models often generalize poorly to data other than what it was trained on [4]. These complications make it difficult to obtain data of sufficient quality by applying traditional natural language processing (NLP) techniques for extraction of medication information from EHRs.

There are specialized methods for misspelling correction and generation in the medical domain, but these often require annotated training data [7], are limited in edit distance, or require language-specific metaphone matching [3], and do not work for misspellings spanning multiple tokens. Sarker and Gonzalez-Hernandez [11] propose a method applying a Levenshtein ratio metric to the semantically similar terms found using word embeddings, which we compare to our own method for misspelling detection.

Coden et al. [2] propose an unsupervised context-based dictionary approach, starting with a set of seed words which are used to identify contexts which are in turn used to recursively expand the dictionary. In this paper, we propose a similar method which employs a bootstrap lexicon, fuzzy

[1] https://github.com/FrankRuis/medical_concept_extraction

matching, a human in the loop, and a way to handle ambiguous dictionary entries. We note that the method is for concept extraction only, entities e.g. describing allergies or medication that is stopped are extracted too and need to be disambiguated in a separate task.

Contributions The contributions of this paper are

- An approach for medication extraction from EHRs that employs a bootstrap lexicon, fuzzy matching, a human in the loop, and a way to handle ambiguous dictionary entries, tailored toward maintaining high precision while iteratively increasing recall, and
- A validation on a real-world data set of 98,200 EHRs of 3,462 patients from a Dutch hospital.

Outlook In Section II, we introduce the data set and what preprocessing we performed. Section III explores the problem by presenting the complications we found in this data for medication extraction. Section IV presents our method. Section V describes the setup and results of our real-world experiments. Finally, conclusions and future work are presented in Section VII.

II. Data & Preprocessing

Our dataset consists of 98,200 EHRs from 3,462 patients from Hospital Group Twente, from January 2001 to February 2020. The characteristics of the reports are as described in the introduction: highly unstructured, ungrammatical, often abbreviated or misspelled, and inconsistent. Some examples of misspellings and abbreviations can be seen in Table I[2]. An example artificial[3], but representative report with its English translation is shown in Figure 1. We manually annotated a small development set of 25 reports, which is used in the main method for optimizing parameters, and a larger test set of 125 reports for evaluation purposes. All reports have been converted to lower case and tokenized using the Natural Language Toolkit [1] treebank word tokenizer. The average report length is 253 tokens (standard deviation 260) and the longest report is 4304 tokens.

III. Problem exploration

In this section, we elaborate on the most important challenges in extracting high quality medication information from textual reports in an EHR. Figure 1 shows an example of an EHR text from out data set with English translation.

A. Weak Structural Clues

Often an organization prescribes standards, conventions, and/or templates to improve quality of the EHRs. We call them *structures* in this paper. Unfortunately, due to changes in structures over time, allowed flexibility in their usage, and imperfect adherence, the structural clues that they give are rather weak complicating a rule-based approach [9].

In our context this is also the case: the EHRs all have a document type or template, but there are 1001 unique document types in total. Moreover, the clinician can choose which template sections to include, in which order, section headings are not standardized, and a section body has no prescribed fixed structure. For example, a medical history section could start with one of the following headings: "med:", "medication", "history", "anamnesis:" (and many more). Furthermore, a report does not necessarily have a medical history section and any other section could contain medication names not mentioned in the history section. Some section headings are used for multiple types of sections, e.g. "history" can indicate a medication history or a more general medical history. The content of a medication section can vary from a simple list of medication names to a multiple paragraph story with occasional mentions of medication names.

Lists of medications can be separated by commas, semicolons, white space, newlines, or a combination thereof. Mode, dosage, duration, and other information may or may not be included after each mention of medication and the format may change in the same list.

B. Misspellings, Typos, and Abbreviations

Table I shows some examples of misspellings, typos, and abbreviations. Some of the complications are caused by a missing or additional space resulting in either two separate tokens being connected or a single token being split up into 2 tokens (e.g., "ascal100mg" or "huma log"). Some more examples are marked in orange in Figure 1 (words marked in green are exact lexicon matches) During data exploration, we found that up to 10% of medication names have such complications. They complicate both medication name and context matching.

C. False Positives and Ambiguity

Some names in the texts are *false positives*, i.e., they match exactly or fuzzily with medication names, but in fact they are not medications. An often occurring example are blood test results, where substances are detected that are also medications. "folic acid" could, for example, be a blood test result but also a medication. Some more examples of this are marked in red in Figure 1. There are also some words that have an ambiguous meaning, such as "stadium" which is a brand name pain relief medication but in Dutch is also a word used to denote a stage of cancer (e.g., "stadium 1 longkanker", in English "stage 1 lung cancer"). Including such terms in a lexicon will result in many false positives if no specific disambiguation technique is applied.

D. Non-Medications in Medication Context

Rule-based approaches typically exploit contextual clues by looking for certain sentence patterns, such as "patient had complication X for which Y". The idea is that Y is usually a medication in this pattern, but unfortunately it could very well also be a treatment such as "physical therapy", a surgery such as "coronary artery bypass", or a non-medical term such as "advice". Also, a medical history or allergies section could for example just contain the word "none", "continue", or

[2]Our dataset only contains Dutch reports, in this paper we show the English translation for readers' convenience.

[3]We do not show original data because of privacy issues.

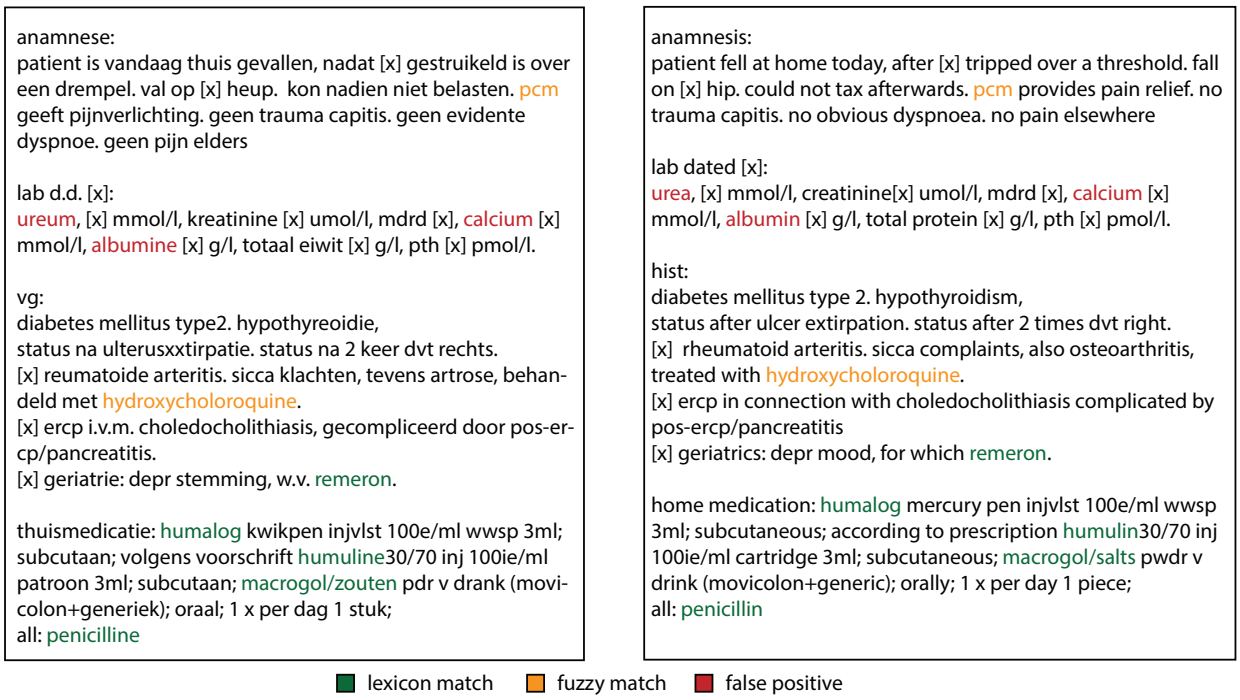

Fig. 1. Anonymized example of (part of) a patient report text with highlighted complications, including English translation on the right.

TABLE I
EXAMPLES OF MISSPELLINGS, ALTERNATE SPELLINGS, AND ABBREVIATIONS (CORRECT SPELLING IN TABLE HEAD)

calci chew d3	**mono cedocard retard**	**hydrochloorthyiazide**	**antibiotica**
clachi chew	cedocard ret	hct	ab
calci-chew/d3	menocedocard retard	hydroloorthiazoide	antibioticakuur
calii hew d3	modocedocard	hydrocholoorthizide	anti bioticum
ca-chew d3	mono-cedocard ret	hydro chloorthiazide	antibiotioca
calcium-chew d3	moncedocard	chloorthiazide	anti-bioticumkuur

"not asked". These non-medications are often hard to reject without human intervention or large amounts of labeled data. Especially the more common words may cause recursive dictionary expansion techniques as also used in [2] to suffer from adding more and more of such non-medications to the lexicon in each iteration.

E. Inter-Annotator Disagreement

There will often be a larger F1-score gap between any two human annotators than there will be between one set of annotations and a state-of-the-art supervised model [13]. These disagreements arise based on what the annotators would consider medications; a strict definition that only includes drug names, or a broader definition that includes anything used to treat symptoms. For example, treatments like "chemotherapy", surgeries like "coronary artery bypass", or distinctions such as "oxygen" vs "air", and even the absence of medication ("none administered") could in some use cases be desirable to extract.

F. Language

Finally, EHRs are usually written in the local language. The texts in our data set are in Dutch. An approach for information extraction should preferably be language-agnostic.

IV. METHOD

Our method relies on a curated lexicon of medication names including their variants, as well as a list of medication contexts. Both, the lexicon and the list of contexts are improved continuously based on errors on the development data set. Sections IV-A to IV-F detail the iterative improvement of lexicon and context list, also depicted as overview in Figure 2, while Section IV-G describes the medication extraction given the final lexicon and contexts.

A. Initial Medication Lexicon Generation

The lexicon is structured as a set of medication names with each a subset of variations and misspellings. Initially the subsets are empty. To seed the dictionary, we scrape initial entries from a trusted source. Our method also works with a small number of seed words, but a rich bootstrap lexicon significantly reduces the reviewing time. We use an online lexicon, which contains all medicines available in the Netherlands and is curated by medical experts, as our initial lexicon[4], containing 10,539 medication names. Other sources

[4]https://farmacotherapeutischkompas.nl, accessed August 2020

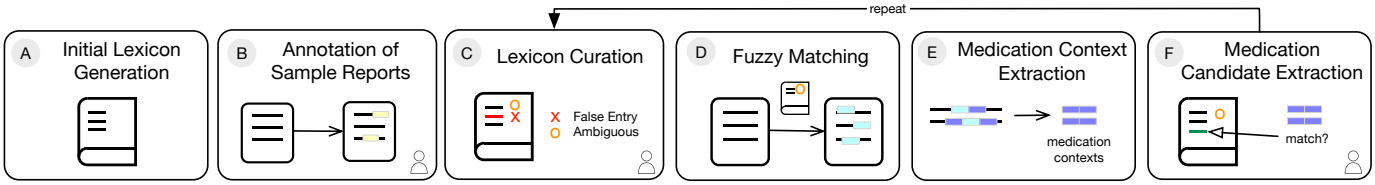

Fig. 2. Overview of the interactive, iterative medication lexicon and context list generation. Steps B, C and F are manually controlled. Candidate and context extraction is repeated (steps C to F) until the quality on the development dataset (generated in step B) is deemed sufficient.

have been considered but were for the most part a more noisy subset of the aforementioned source.

B. Annotation of Sample Reports

To annotate the development set, we used the initial lexicon to pre-annotate the reports, and then manually corrected errors. The pre-annotation significantly reduced the total annotation time. The initial annotation does not require domain knowledge, since missing instances can be corrected later (cf. section IV-C).

C. Lexicon Curation

As described in section III-C, one of the main issues with a dictionary approach is that ambiguous entries will cause false positives. To alleviate this, we track ambiguous lexicon entries. We use the development set to identify all lexicon entries that produce false positives, which will fall in one of three categories: (i) The annotation is an error in the lexicon and is removed from the lexicon. (ii) It is a manual annotation error and the annotation is corrected. (iii) It is an ambiguous entry in the lexicon and will be marked.

Annotations that correspond to ambiguous lexicon entries will be rejected when they occur in a context whose partial match score is below a certain threshold (cf. section IV-E). A suitable value for this threshold can be found by e.g. increasing it until the evaluation on the development set produces no false positives or by setting it close to the average value of unambiguous lexicon entry contexts. For our experiments we did the latter, resulting in a threshold of 0.01.

D. Fuzzy Matching

Since a significant portion of the medication names are abbreviated or misspelled, we employ fuzzy matching to maximise the amount of lexicon entries available for the next steps. Our dataset has almost 200,000 unique tokens and the initial lexicon has thousands of entries. This makes edit distance calculations for distances greater than 1 quite computationally intensive. And while many misspellings are within 1 edit distance of a lexicon entry, there are also many non-medication tokens within 1 edit distance of lexicon entries.

For these reasons we use a Term Frequency - Inverse Document Frequency (TF-IDF) weighted cosine similarity approach which is common in information retrieval. Since this amounts to a sparse matrix dot product and we are only interested in the closest match, we use an efficient implementation of a sparse dot function[5], developed by the ING data science team. All unique tokens are first converted to character-level 3-grams, which are then converted to a matrix of TF-IDF features. The cosine similarity threshold for matches is fine-tuned such that the new entries produce no false positives on the development set.

The fuzzy matches are the only part of the algorithm that results in unreviewed lexicon entries and as such the most likely part to generate false positives. Because many misspellings can also be caught in the review step, the threshold can be set to a value that sacrifices some recall for perfect precision on the development set, which in our experiments was 0.85.

For multi-token matches we first calculate the pointwise mutual (PMI) information for all neighboring tokens:

$$PMI(x, y) = log \frac{p(x, y)}{p(x)p(y)}$$

Where $p(x, y)$ is the probability that tokens x and y occur together, which is divided by $p(x)p(y)$, the probability that the tokens occur separately. Tokens with a PMI exceeding a certain threshold are combined. The fuzzy matching process above is then repeated for these token combinations and the multi-token lexicon entries.

Fuzzy matches are added as a sub-element of the original lexicon entry and excluded in future runs of the fuzzy matching process. In this way different spellings have a single parent entry and future runs will not recursively match to increasingly distant tokens. Fuzzy matches of ambiguous lexicon entries are marked as ambiguous as well.

Additionally, we compare our method to Sarker and Gonzalez-Hernandez [11] by training word2vec [8] word embeddings on our own dataset and tuning the parameters such that no false positives are generated on the development set.

E. Medication Context Extraction

For each element in the lexicon, we find matches in the EHRs and extract an n-token context window. For each element x_i in a tokenized sentence X, the context C_i is defined as:

$$C_i = \bigcup_{k=-n}^{n} \{x_{i+k} | k \neq 0\}$$

Where we wrap to at most one line above or below and use the empty string when that line is empty or another

[5]https://github.com/ing-bank/sparse_dot_topn, accessed August 2020

sentence boundary is reached. The 3-token context window of "humalog" in Figure 1 would be: ("home", "medication", ":", "mercury", "pen", "injvlst").

All numbers and known medication names are masked to reduce the number of unique contexts (i.e. known medication tokens in the lexicon are replaced with the token 'M' and numeric tokens are replaced with the token 'D').

For each context we calculate a certainty score, which is the proportion of occurrences where the context surrounds a known lexicon entry as opposed to a token outside of the lexicon. If the certainty is lower than a certain threshold θ_1, i.e., there are too many instances where the context surrounds a non-medication token, the context is rejected. For our data 0.25 was a sufficient threshold, this can be increased to ease the workload for the human reviewer or lowered to find more elusive matches. Contexts that only surround known lexicon entries are excluded for efficiency, since they can not lead to new candidates.

Each context is assigned a partial match score s

$$s = \frac{1}{n} \sum_{i=0}^{n-1} \frac{c_i}{g_i},$$

where n is the context window size, c_i is the number of times the token at position i occurs in the same position in a medication context, and g_i is the number of times the token at position i occurs in the same position in any context.

Contexts that do not surround a lexicon entry, but have a partial match score that exceeds a certain threshold θ_2, are added to the list of medication contexts. In this way we can capture candidates that e.g. share a medication context except for one unique misspelled token. For our experiments we used a threshold of 0.25.

The thresholds θ_1 and θ_2 have been based on preliminary experiments. A higher threshold means less work for the reviewer but decreased recall. In our case the thresholds filtered out a significant amount of false positives and kept the manual reviewing effort low.

F. Medication Candidate Selection

Existing lexicon entries are added to an exclusion list, such that they are not shown in the manual review for extending the lexicon. New candidates are added if their context matches a context from the context list, but do not belong to an existing lexicon entry. To identify multi-token candidates, we also evaluate with up to 3 tokens in the center of each context. This number was chosen since very few medication names exceed 3 tokens in length.

Candidates are ranked based on the amount of occurrences in the entire dataset, from highest to lowest. They are shown to reviewer in batches on a graphical user interface where the reviewer can mark candidates as a regular or ambiguous match. If the reviewer missed an ambiguous entry here, it will most likely be corrected when returning to the lexicon curation step where the newly accepted candidates are evaluated on the development set. Unmarked candidates are rejected and saved to the exclusion list, to prevent them from showing up in the review queue again.

The lexicon is extended with the marked candidates and the process can repeat until no new candidates can be found or a desired accuracy is reached.

G. Final Medication Extraction

Once the dictionary expansion process is complete we are left with the final lexicon, ambiguous set, and the occurrence statistics needed to calculate the partial match score. Tokens in unseen samples will then be tagged according to a regular dictionary method using the final lexicon, with the added step of rejecting tokens that are contained in the ambiguous set when their partial match score does not exceed the threshold, as described in section IV-C.

The final output of the algorithm is the extended lexicon and the occurrence statistics needed to calculate the partial match score. One can optionally save the rejected candidates too so the algorithm can be reran later on new reports.

V. Results & Discussion

A. Evaluation

The results are evaluated by precision, recall, and F1-score. When calculating these values, each token is counted only once per report, e.g. if a true positive medication name is mentioned 6 times in the same report and a false negative (FN) only once, the combined recall for this report is counted as 50%. The reason for this approach is that for our application we want to extract a timeline of medication information. Multiple mentions of the same medication in the same report do not offer more information for such a timeline, and since a dictionary approach catches all mentions at once it might skew the results to appear more accurate than they really are. For comparison purposes we offer the final results calculated in a more standard way, where all occurrences are counted, as well.

All execution timings have been done on a virtual machine running Ubuntu, with an Intel Xeon E-2124 4 core processor @ 3.30GHz, and 12 GB RAM.

Table II shows the results on the test set after each stage of the algorithm. Column A shows the results when including the fuzzy matching step, column B without.

The initial lexicon contains some false positives such as the blood test result "ureum", or the pain medication "centrum" which can also refer to e.g. a city centre. The false negatives are mainly alternate names such as brand names or chemical compounds, misspellings, or abbreviations.

At the end of the second iteration of the algorithm we already have over 96% recall on the test set, while maintaining 100% precision. The parameters have been tuned to produce 0 false positives on the development set, and this has been successfully carried over to the test set. In total we identified 955,762 medication names in all 98,200 reports, around 10 per report on average.

TABLE II
RESULTS ON THE TEST SET AFTER EACH STAGE OF THE ALGORITHM, WITH OUR FUZZY MATCHING METHOD (A), SARKER AND GONZALEZ-HERNANDEZ' METHOD [11] (B), AND WITHOUT FUZZY MATCHING (C)

	A			B			C		
Stage	**precision**	**recall**	**F1-score**	**precision**	**recall**	**F1-score**	**precision**	**recall**	**F1-score**
Bootstrap lexicon	0.895	0.689	0.779	0.895	0.689	0.779	0.895	0.689	0.779
Marked ambiguous entries	1.0	0.689	0.816	1.0	0.689	0.816	1.0	0.689	0.816
Fuzzy matching	1.0	0.775	0.873	1.0	0.768	0.868	-	-	-
Review 1	1.0	0.923	0.960	1.0	0.874	0.937	1.0	0.881	0.937
Review 2	1.0	0.968	0.984	1.0	0.962	0.954	1.0	0.912	0.954
Review 2 (alternate)[6]	1.0	0.976	0.988	1.0	0.971	0.958	1.0	0.919	0.958

B. Fuzzy Matching

With the sparse dot function it takes less than one second to compare all 1.2 million combinations of tokens. The final lexicon contains over 4,000 unique fuzzy matches.

The multi-token fuzzy matching step significantly improves the detection of medication names that span multiple tokens and single token medication names that have been spelled with an erroneous space. All variations seen in Table I are automatically identified in the fuzzy matching step for example, except for the acronyms (hct, pcm) which are identified in the human review step.

As shown in Table II column C, omitting the fuzzy matching step led to a 5.61% decrease in recall on the test set, in addition to a large increase in work for the human reviewer as there are now at least 4,000 more candidates to manually accept.

The Sarker and Gonzalez-Hernandez [11] misspelling generation method performs on par with our own. Combining our method with theirs by taking the union of the misspellings produced by each method results in a slightly higher final recall of 0.974 (not reported in table II). However, since additional parameters need to be tuned and a higher chance of not reviewed false positives, we suggest using just one of the methods. The main differences are that our method can find misspellings which are too rare to be embedded close to its correct spelling and misspellings spanning multiple tokens, while their method can afford to choose a lower acceptance threshold because it will only compare semantically similar tokens.

C. Human in the Loop

Experiments in applying the algorithm in an unsupervised manner, without reviewer, resulted in exponential loss of accuracy on even the most conservative parameter thresholds. This is mainly due to the large number and variety of common words such as "Unknown" or "None administered" which occur in highly confident medication contexts, and can not be filtered out without time-intensive rule-based methods tailored to the dataset or large amounts of annotated training data (see also Section VI).

The human-in-the-loop approach ensured that all ambiguous entries were marked as such and all false positives were rejected, resulting in a precision of 100%. Cases such as those mentioned above are easily filtered out. Candidates that usually cause inter-annotator disagreement, as described in Section III-E, can be accepted or rejected at the reviewer's discretion.

Using the GUI, it took a reviewer without domain knowledge 5 minutes to review 200 candidates, with most time spent looking up if a candidate really is a medication name. A domain expert could likely review close to as fast as they can read. Without optimization one run of the algorithm takes a couple of minutes on our data set, which could easily be improved since all actions are highly parallelizable.

The certainty score (cf. Section IV-E) does a good job at ensuring most candidates are actually medication as to not waste the reviewer's time, while the sorting on occurrence counts ensures that the candidates that would lead to the most new contexts are reviewed first. Table III shows the amount of candidates that have been reviewed and the resulting test F1-score. We stopped after 1000 candidates in review 2 since there weren't many matches after that. The table shows the F1-score if we would have continued until the end of the candidate list, assuming none of the matches would have resulted in false positives. That last part was done simply by checking which remaining false negatives from the test set were present in the unexplored part of the candidate list.

TABLE III
AMOUNT OF ENTRIES REVIEWED VS TEST F1-SCORE.

Stage	**Candidates Reviewed**	**Test F1-score**
End review 1	250	0.960
Review 2	250	0.973
Review 2	500	0.979
Review 2	750	0.981
End review 2	1000	0.984
Check for remaining FNs	2000	0.985
Check for remaining FNs	4000	0.988
End of candidate list	7614	0.989

D. False Negatives

The remaining false negatives usually fall in one of 3 categories (in addition to a unique context):

- Highly variable spellings of a combination of 2 or more medications. (e.g. bupikenacort or kena/bupi, bupivacaine and kenacort are both in the final lexicon)
- Short tokens with 1 edit distance spelling mistakes which would introduce a large amount of false positives if the fuzzy matching threshold is set low enough to catch them. (e.g. volufen instead of voluven)

[6]Results calculated in the standard way as described in section V-A

- Relatively new and/or rare treatments. (e.g. bacteriophages)

VI. Comparison with Coden et al.

We initially intended to implement Coden et al. [2] for a numeric comparison, as their method is closely related to ours, but some complications arose that hinder a fair comparison.

Our data contains many different instances of common words in high confidence medication contexts, such as "medication: continue." or "patient was given advice.". Because in all other instances the context does surround a valid medication, even the most strict parameters won't filter such a context out. As such, a completely unsupervised approach will become exponentially worse as more common false positives are added to the lexicon and used to find new contexts after each iteration.

Similarly, the lack of handling ambiguous medication will result in a final lexicon that contains every substance that could be found in a blood test, which adds 10+ false positives to every report containing such test results.

The lack of fuzzy matching also means that a lot of medication will be missed, as shown in Table II.

These complications are mainly a result of the unstructured nature of our dataset, making comparison to our results unfair as their method was not developed for such data.

VII. Conclusions

We propose a robust method for extracting a medication history from a patient's health records. Central to the approach is to obtain a high quality lexicon of medication names and their alternative forms from a large corpus of health records. The method is efficient, language-agnostic, and works accurately on highly unstructured data with ambiguous entries. The human-in-the-loop approach ensures that the final lexicon is of high quality and complies with the user's opinions as opposed to be subject to disagreements with annotations from an external source. The investment of human attention is limited: about 5 min for reviewing 200 candidates for a non-informed reviewer; less for a domain expert. In our experiments, we reviewed 1250 candidates, which brought the recall of the extracted medication names up to 96.8% on the test set. Thanks to the human involvement, precision was maintained at 100%.

Future work could include applying and evaluating the method on other tasks such as extracting other factual information such as co-morbidity, or using the output annotations to train a weakly supervised model.

References

[1] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. Beijing: O'Reilly Media Inc., 2009.

[2] A. Coden, D. Gruhl, N. Lewis, M. Tanenblatt, and J. Terdiman, "SPOT the Drug! An Unsupervised Pattern Matching Method to Extract Drug Names from Very Large Clinical Corpora," in *2012 IEEE Second International Conference on Healthcare Informatics, Imaging and Systems Biology*, Sep. 2012, pp. 33–39.

[3] P. Fivez, S. Šuster, and W. Daelemans, "Unsupervised Context-Sensitive Spelling Correction of Clinical Free-Text with Word and Character N-Gram Embeddings," in *BioNLP 2017*. Vancouver, Canada,: Association for Computational Linguistics, Aug. 2017, pp. 143–148. [Online]. Available: https://www.aclweb.org/anthology/W17-2317

[4] J. M. Giorgi and G. D. Bader, "Towards reliable named entity recognition in the biomedical domain," *Bioinformatics*, vol. 36, no. 1, pp. 280–286, Jan. 2020, publisher: Oxford Academic.

[5] S. Henry, K. Buchan, M. Filannino, A. Stubbs, and O. Uzuner, "2018 n2c2 shared task on adverse drug events and medication extraction in electronic health records," *Journal of the American Medical Informatics Association*, vol. 27, no. 1, pp. 3–12, Jan. 2020, publisher: Oxford Academic.

[6] Y. Kim and S. M. Meystre, "Ensemble method-based extraction of medication and related information from clinical texts," *Journal of the American Medical Informatics Association: JAMIA*, vol. 27, no. 1, pp. 31–38, 2020.

[7] K. H. Lai, M. Topaz, F. R. Goss, and L. Zhou, "Automated misspelling detection and correction in clinical free-text records," *Journal of biomedical informatics*, vol. 55, pp. 188–195, 2015.

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[9] S. Pathak, J. van Rossen, O. Vijlbrief, J. Geerdink, C. Seifert, and M. van Keulen, "Post-structuring radiology reports of breast cancer patients for clinical quality assurance," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, May 2019.

[10] J. Patrick and M. Li, "High accuracy information extraction of medication information from clinical notes: 2009 i2b2 medication extraction challenge," *Journal of the American Medical Informatics Association*, vol. 17, no. 5, pp. 524–527, Sep. 2010.

[11] A. Sarker and G. Gonzalez-Hernandez, "An unsupervised and customizable misspelling generator for mining noisy health-related text sources," *Journal of Biomedical Informatics*, vol. 88, pp. 98–107, Dec. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1532046418302168

[12] C. Tao, M. Filannino, and O. Uzuner, "Prescription extraction using CRFs and word embeddings," *Journal of Biomedical Informatics*, vol. 72, pp. 60–66, Aug. 2017.

[13] O. Uzuner, I. Solti, F. Xia, and E. Cadag, "Community annotation experiment for ground truth generation for the i2b2 medication challenge," *Journal of the American Medical Informatics Association*, vol. 17, no. 5, pp. 519–523, Sep. 2010, publisher: Oxford Academic.

[14] M. Van Keulen, J. Geerdink, G. Linssen, R. Slart, and O. Vijlbrief, "Exploiting natural language processing for improving health processes," in *Proceedings of the 7th International Symposium on Data-Driven Process Discovery and Analysis (SIMPDA 2017)*, ser. CEUR Workshop Proceedings, P. Ceravolo, M. van Keulen, and K. Stoffel, Eds. CEUR, 12 2017, pp. 145–146.

[15] H. Xu, S. P. Stenner, S. Doan, K. B. Johnson, L. R. Waitman, and J. C. Denny, "MedEx: a medication information extraction system for clinical narratives," *Journal of the American Medical Informatics Association*, vol. 17, no. 1, pp. 19–24, Jan. 2010, publisher: Oxford Academic.