QuickUMLS: a fast, unsupervised approach for medical concept extraction

Luca Soldaini
Information Retrieval Lab
Georgetown University
Washington, DC, USA
luca@ir.cs.georgetown.edu

Nazli Goharian
Information Retrieval Lab
Georgetown University
Washington, DC, USA
nazli@ir.cs.georgetown.edu

ABSTRACT

Entity extraction is a fundamental step in many health informatics systems. In recent years, tools such as MetaMap and cTAKES have been widely used for medical concept extraction on medical literature and clinical notes; however, relatively little interest has been placed on their scalability to large datasets. In this work, we present QuickUMLS: a fast, unsupervised, approximate dictionary matching algorithm for medical concept extraction. The proposed method achieves similar precision and recall of state-of-the-art systems on two clinical notes corpora, and outperforms MetaMap and cTAKES on a dataset of consumer drug reviews. More importantly, it is up to 135 times faster than both systems.

CCS Concepts

•Applied computing → Health care information systems;

Keywords

Concept extraction; Health informatics; Biomedical mining

1. INTRODUCTION

One of the cornerstones of many health informatics systems is the identification of medical entities in unstructured documents. A system designed to address this task receives a document as input and returns a set of spans in the documents, as well as identifiers of medical concepts associated with each span (such as concepts in the UMLS metathesaurus[1]). The concepts can then be used in, among others, information retrieval [6, 10], question answering [14], and clinical events detection and parsing [3] tasks.

Many have focused on increasing the precision and recall of medical information extraction systems; yet, relatively little attention has been dedicated to their efficiency. This limits their applicability to large datasets. In this work, we introduce a system that relies on approximate matching to terms in UMLS to extract medical concepts from unstructured text. Our implementation is able to extract concepts using the entire English subset of UMLS from a document of approximately 1000 tokens[2] within 500 to 1000 ms, depending on the threshold used for approximate string matching. Furthermore, since its inverted index is stored on disk, it requires a modest amount of memory (most of which is used by a shallow tagger). An implementation of our system in Python is freely available to researchers[3].

Concept extraction pipelines such as MetaMap [2] and cTAKES [12] are widely used in research and industry applications. The former uses a shallow parser to generate candidate phrases; then, for each candidate phrase, many lexical variations are generated; finally, each phrase is scored based on their distance to concepts in UMLS. A word disambiguation tool is also used to favor concepts that are semantically consistent with surrounding text. Similarly, cTAKES matches candidate phrases (as well as their permutations and lexical variations) with concepts in UMLS and a list of concepts maintained by Mayo Clinic.

Many other solutions — often designed for specific scenarios — have been introduced in recent years. For example, Zhou et al. [17] proposed *MaxMatcher*, a fuzzy matching approach to extract biomedical concepts. Like our system, *MaxMatcher* performs approximate dictionary matching against UMLS. Their system considers a token as minimum edit distance between two expressions; Thus, unlike QuickUMLS, it does not capture variations of terms (e.g., "tumor" vs "tumour") unless they are present in UMLS. Delbecque and Zweigenbaum [4] introduced a system that attempts at matching noun phrases to concepts in UMLS. Terms in noun phrases are replaced by their lemma before being matched with concepts; concepts that share only a subset of terms with a noun phrase are still considered unless they are significantly longer or shorter than the noun phrase. This approach, while suited for the drug reviews dataset — where 95% of the annotations are noun phrases — would not be effective for the clinical notes datasets, where approximately only half of the concepts (48% for i2b2 dataset, 49% for THYME corpus) are noun phrases. Abacha and Zweigenbaum [1] introduced *MeTAE*, a rule-based concept extraction algorithm. *MeTAE* employs sentence chunking before using MetaMap to obtain concepts. When testing MetaMap

[1]The Unified Medical Language System is a collection of medical thesauri maintained by the US National Library of Medicine.

MedIR '16 Pisa, Italy

DOI: xxxxxxxxxxx

[2]A token is any sequence of alphanumeric characters or punctuation symbols delimited by whitespace.

[3]`https://github.com/Georgetown-IR-Lab/QuickUMLS`

against the method proposed in this paper, we chunk the input text into sentences before processing it with MetaMap. Divita et al. [5] proposed a system that performs exact concept mapping of terms in the UMLS metathesaurus. Their system attempts to match the maximum number of terms within a sliding window; if no match is found, the last term in the window is dropped. Compared to our system, their solution does not support approximate matching of concepts, thus making it less flexible.

In summary, the contribution of this work is threefold: (*i*) we introduce a system to perform UMLS concept extraction on unstructured documents; the proposed method is significantly faster than other state-of-the-art systems while achieving comparable accuracy; (*ii*) we evaluate its performance on three different datasets ranging from clinical notes to drug reviews written by laypeople; (*iii*) we release an implementation of the system for other researches to use.

2. METHODOLOGY

The problem of extracting concepts from unstructured documents can be formalized as follows. Let a dictionary $\mathcal{S}$ be a set of strings, $\mathcal{C}$ collection of concepts, $C : \mathcal{C} \rightarrow \mathcal{S}$ a map that associates a concept in the collection to one or more strings in the dictionary. Given a document d, which we represent as a sequence of tokens $\{d_1, \ldots, d_n\}$, a similarity function *strsim*, and a similarity threshold $\alpha \in [0, 1]$, a concept extraction algorithm returns the following set:

$$\{(d_{ij}, c_k) \mid \exists s_h \in C(c_k) \text{ s.t. } strsim(d_{ij}, s_h) \geqslant \alpha\} \quad (1)$$

where d_{ij} represents a sequence of tokens in d and $s_h \in \mathcal{S}$ is a string representing concept $c_k \in \mathcal{C}$. In this work, we require that no overlapping concepts should be extracted; this is due to the fact that no overlapping concepts are present in the datasets used for evaluation. The implementation of our system includes an option to return overlapping concepts.

The problem of approximate dictionary matching for a string can be formally stated as follows: given a target string x, a threshold α, a dictionary $\mathcal{S}$, and a similarity function *strsim*, we wish to find the subset $\mathcal{Y}_{x,\alpha} \subseteq \mathcal{S}$ such that

$$\mathcal{Y}_{x,\alpha} = \{y \in \mathcal{S} \mid strsim(x, y) \geqslant \alpha\} \quad (2)$$

Many functions can be used to estimate the similarity of strings x and y; in this paper we use Jaccard similarity:

$$Jaccard(x, y) = |x \cup y| \ / \ |x \cap y| \quad (3)$$

A naïve solution of the approximate dictionary matching problem is to compute the similarity of each string in $\mathcal{S}$ to the target string x. This approach has complexity $O(|\mathcal{S}|)$; thus, it is computationally expensive in applications where $\mathcal{S}$ is large. This aspect makes the naïve approach unfeasible, as the English subset of UMLS contains more than 6 million strings. Our approach leverages CPMerge, an algorithm for approximate dictionary matching introduced by Okazaki and Tsujii in [9].

2.1 CPMerge

Rather than computing the similarity with every string in the vocabulary, CPMerge obtains the subset $\mathcal{Y}_{x,\alpha}$ by representing strings as a set of features, and then identifying those strings in $\mathcal{S}$ that have more than τ features in common with the target string x. CPMerge represents the dictionary $\mathcal{S}$ as an inverted index that associates each feature with the strings containing such feature; thus, the problem of the approximate dictionary matching is equal to finding a solution to the τ-*overlap join* problem [11] on the posting lists of the features of target string x (i.e., the list of strings associated with each feature [7]), as shown in [9].

In detail, CPMerge determines $\mathcal{Y}_{x,\alpha}$ as follows: first, it considers character trigrams as features; for example, the string $x =$ *"tumor"* is associated with the following set of features: $X = \{$`##t`, `#tu`, `tum`, `umo`, `mor`, `or#`, `r##`$\}$, where the pound sign denotes the beginning or the end of a string.

Then, it computes the minimum and maximum size of the feature set Y of any string $y \in \mathcal{S}$ that could have at least τ features in common with x. When Jaccard similarity is used, $\min |Y| = \lceil \alpha \cdot |X| \rceil$ and $max|Y| = \lfloor |X|/\alpha \rfloor$. The minimum number of overlapping features τ also depends on the similarity function used; in our case, $\tau = \lceil (\alpha \, (|X| + |Y|) \, / (1 + \alpha) \rceil$. For example, for $x =$ *"tumor"*, $|X| = 7$; thus, any string $y \in \mathcal{S}$ such that $Jaccard(x, y) \geqslant 0.7$ must have between $\lceil 0.7 \cdot 7 \rceil = 5$ and $\lfloor 7/0.7 \rfloor = 10$ features. We remand the reader to [9] for more details on how $min|Y|$, $max|Y|$, and τ are derived from Equations 2 and 3.

Once the minimum and maximum lengths are determined, CPMerge obtains, for each $l \in \{min|Y|, \ min|Y| + 1, \ \ldots, \max |Y|\}$, the set $\mathcal{Y}_{x,\alpha,l}$ of strings of length l that have more than τ features in common with x. This is done by joining the posting lists of each feature in X, and keeping those strings that appear in more than τ lists. For efficiency reasons, each posting list is partitioned in sets containing all strings of the same length; thus, CPMerge can join the subset of each posting lists containing strings of length l without visiting the entire list. The set of all approximate dictionary matching of string x is obtained by considering the union of $\mathcal{Y}_{x,\alpha,l}$ for all acceptable values of l.

2.2 QuickUMLS

Given a document d of length n, a similarity threshold α, and a window size w, QuickUMLS efficiently generates, for each token $d_i \in d$, all possible sequences of tokens $d_{ij} = \{d_i, \ldots, d_j\}, j \in \{i, \ldots, i + w - 1\}$[4]. Then, a set of heuristics is used to determine whether d_{ij} is a valid sequence of tokens; if that is the case, CPMerge is used to identify strings in $\mathcal{S}$ that are similar to d_{ij}. Once the subset of all possible matching strings $\mathcal{Z}_{d,\alpha} = \bigcup_{d_{ij}} \left(\mathcal{Y}_{d_{ij},\alpha}\right)$ is determined, QuickUMLS selects the most appropriate subset $\mathcal{Z}'_{d,\alpha}$ of strings such that there is no overlap between the set of extracted concepts.

In detail, QuickUMLS proceeds as follows. First, it tokenizes d and obtains part of speech tags for each token. SpaCy[5] was used to accomplish both tasks. Then, $\forall i, j \in \{1, \ldots, n\}, j \leqslant (i + w - 1)$, it generates all possible valid sequence of tokens. A sequence of tokens d_{ij} is valid if:

- d_{ij} contains at most w tokens. A token could be a word or punctuation; we allow punctuation to be part of a sequence of tokens because, in the medical domain, punctuation is often part of the name of a drug, disease, or treatment (e.g., "*Lantus (Insulin Glargine)*"). Numbers are also allowed to appear in valid sequences, as long as they are not the only token in the sequence, so that additional information such as particular dosage of drugs (e.g., "*Tylenol 325mg Tablet*") are captured.

[4]if $(i + w - 1) > n$, then $j \in \{i, \ldots, n\}$

[5]`v.0.100.7`, `https://spacy.io/`

- d_{ij} does not span across sentences[6].
- The first or the last tokens of the sequence (d_i and d_j, respectively) are not a conjunction, an adposition, or a determiner as determined by the extracted part-of-speech tags. Such tokens are kept as long as they appear within a sequence; this was determined to be a good strategy for handling strings such as "*difficulty in walking*".
- The first token in the sequence (d_i) is not punctuation.
- d_{ij} is not a stop word or a number if d_{ij} contains only one token (i.e., $i = j$).

Then, CPMerge is used to find strings in $\mathcal{S}$ whose similarity to d_{ij} is greater or equal to α and add them to $\mathcal{Z}_{d,\alpha}$. Given the set $\mathcal{Z}_{d,\alpha}$ of all possible sequences in d that can be associated with concepts in $\mathcal{C}$, QuickUMLS selects a subset of string with no overlap that maximizes the sum of the similarity scores with token sequences in d. In detail, it greedily chooses sequences to include in $\mathcal{Z}'_{d,\alpha}$ based on their similarity to a string in $\mathcal{S}$. In other words, for two overlapping sequences d_{ij} and d_{pq}, d_{ij} is added to $\mathcal{Z}'_{d,\alpha}$ if for all $s \in \mathcal{S}$, $strsim(d_{ij}, s) > strsim(d_{pq}, s)$. In case of ties (that is, there exists $s \in \mathcal{S}$ such that $strsim(d_{ij}, s) = strsim(d_{pq}, s)$), the longer sequence is chosen.

3. EVALUATION

In this section, we compare the proposed approach with MetaMap[7] and cTAKES[8]. We report the performance of the three systems in terms of precision, recall, and F-1; furthermore, we analyze the running time of each pipeline. Each system is tested against three datasets: two corpora of clinical notes and a collection of drug reviews. The first two were chosen because both cTAKES and MetaMap are known to perform well on clinical notes, although the latter could be effected by lack of proper syntactical structure.

We also examined the performance of QuickUMLS on laypeople-generated content, as tools designed for concept extraction on clinical notes are known to struggle on it. In particular, we were interested in understanding whether the approximate dictionary match approach introduced in this paper could address syntactic and spelling mistakes often found in reviews written by consumers.

All experiments were carried out on a workstation with two AMD Opteron 4386 CPUs, 32GB of RAM, and a SSD.

3.1 Datasets

3.1.1 2010 i2b2/VA Challenge

The 2010 i2b2/VA Challenge dataset [15] is a collection of clinical notes from the Veterans Affairs (VA) electronic medical records system. We use a subset of 169 reports that have been annotated with medical concepts.

Notes in the subset have a mean length of 1040 tokens (maximum 4460, minimum 123, median 990); on average, each document contains 99 medical concepts (maximum 427, minimum 2, median 83). As suggested in [8], we considered UMLS concepts from 16 semantic types that are typically associated with the *four aspect of the medical decision criteria* (namely symptoms, diagnostic tests, diagnoses, and treatments); that is, they represent necessary information health practitioners need to assist their patients.

We will refer to this collection as the "i2b2 dataset" throughout the reminder of the paper.

[6]SpaCy determines sentence boundaries from the syntactic dependency parse tree.

[7]`v.2016`, `UMLS 2015AB` release, `NegEx` processing enabled. Rather than using MetaMap's chunker, we preprocess input text with SpaCy, as it is much faster.

[8]`v.3.2.2`, `FastUMLSProcessor` pipeline.

Method		*Prec*	*Rec*	*F-1*	*ms/doc*
MetaMap		0.49*	0.48*	0.48*	19,295*
cTAKES		**0.71**	0.53*	0.61	3,852*
QuickUMLS	$\alpha = 0.6$	0.50*	**0.75**	0.60	1,594*
	$\alpha = 0.7$	0.60*	0.66*	**0.63**	680*
	$\alpha = 0.8$	0.63*	0.60*	0.61	332*
	$\alpha = 0.9$	0.64*	0.56*	0.60	193*
	$\alpha = 1.0$	0.67*	0.54*	0.60	**143**

Table 1: Results for the i2b2 dataset. cTAKES outperforms QuickUMLS in precision, but QuickUMLS has better recall. QuickUMLS is 2.5 to 135 times faster than MetaMap or cTAKES. * indicates statistically significant differences from best value (Welch's t-test, $p < 0.05$).

3.1.2 THYME

The THYME corpus [13] consists of 1254 de-identified clinical reports from a large US healthcare provider (Mayo Clinic). The reports summarize the interactions between physicians and patients in the oncology department. Each note was annotated with the following entities: temporal events, temporal relations, and clinical concepts. In this paper, we use the clinical concepts to evaluate the performance of the proposed method.

Clinical reports in THYME have a mean length of 1035 tokens (max 4423, min 114, median 836); on average, each document contains 172 medical concepts (max 779, min 15, median 127). As for the i2b2 dataset, we considered the subset of UMLS concepts proposed in [8].

3.1.3 Drug Reviews

We also evaluate the proposed system on a corpus of consumer-generated reviews for five commonly used breast cancer drugs: *Anastrozole*, *Exemestane*, *Letrozole*, *Raloxifene*, and *Tamoxifen*. This dataset was introduced by Yates and Goharian [16]; it contains 2500 reviews from `askapatient.com`, `drugs.com`, and `drugratingz.com`. We use a subset of 250 reviews annotated for mentions of adverse drug reactions. Reviews in this dataset are substantially shorter than clinical notes in the i2b2 and THYME corpora, having an average length of 131 tokens (max 580, min 24, median 114). Furthermore, they contain, on average, just 2 mentions of adverse drug reactions (max 9, min 1, median 2). For this dataset, we considered UMLS concepts associated with the following semantic types: "*Sign or Symptom*", "*Disease or Syndrome*", "*Finding*", and "*Neoplastic Process*".

3.2 Results

The performances of QuickUMLS, cTAKES, and MetaMap on the datasets introduced in Section 3.1 are shown in Table 1, 2, and 3. The size of the window w used by QuickUMLS was set to 5 after empirical evaluation; the effect of different values for the similarity threshold α are studied. Regardless of the value of α chosen, the running time of QuickUMLS is dominated by the approximate matching subroutine.

As shown in Table 1, cTAKES outperforms our method in terms of precision on the i2b2 dataset. This is an expected result, as cTAKES is optimized to process clinical narratives. On the other hand, QuickUMLS achieves better recall than any other method when $\alpha = 0.6$, outperforming

Method		*Prec*	*Rec*	*F-1*	*ms/doc*
MetaMap		0.71*	0.53*	0.61*	15,935
cTAKES		**0.89**	0.55*	0.68*	3,765*
QuickUMLS	$\alpha = 0.6$	0.68*	**0.77**	**0.72**	1,536*
	$\alpha = 0.7$	0.78*	0.67*	**0.72**	646*
	$\alpha = 0.8$	0.83*	0.61*	0.70†	340*
	$\alpha = 0.9$	0.85*	0.57*	0.68*	174*
	$\alpha = 1.0$	0.87*	0.55*	0.67*	**141**

Table 2: Results for the THYME corpus. cTAKES achieves the best precision, QuickUMLS the best recall and substantially better throughput than MetaMap or cTAKES. * indicates statistically significant differences from best value (Welch's t-test, $p < 0.05$).† indicates statistical significance from $\alpha = 0.6$ but not $\alpha = 0.7$.

Method		*Prec*	*Rec*	*F-1*	*ms/doc*
MetaMap		0.12*	0.16*	0.14*	1,774*
cTAKES		0.16*	0.37*	0.22*	301*
QuickUMLS	$\alpha = 0.6$	0.16*	**0.6**	0.25*	116*
	$\alpha = 0.7$	0.38*	0.51	0.44*	57*
	$\alpha = 0.8$	0.43	0.51	0.47	32*
	$\alpha = 0.9$	**0.47**	0.50	**0.48**	22
	$\alpha = 1.0$	**0.47**	0.45*	0.46	**18**

Table 3: Results for the drug reviews dataset. QuickUMLS outperforms both cTAKES and MetaMap on content generated by laypeople. * indicates statistical significance from best value (Welch's t-test, $p < 0.05$).

cTAKES in terms of F-1 score, although the difference is not statistically significant. Furthermore, we observe that QuickUMLS exhibits comparable performances to cTAKES when $\alpha = 1.0$ (*Prec*: -6% , *Rec*: $+2\%$, *F-1*: $+3\%$), yet it is 26 times faster. Surprisingly, MetaMap performs poorly; we attribute this to the fact that MetaMap is more sensible to the lack of syntactic structure and the use of abbreviations.

The results for the THYME corpus (Table 2) closely resemble those obtained on the i2b2 dataset: cTAKES achieves the best precision (0.89 vs 0.87), while QuickUMLS shows a better recall for smaller values of α (0.77 vs 0.55). Moreover, when $\alpha = 1.0$, QuickUMLS achieves nearly identical performances to cTAKES (-2% precision, statistically significant; identical recall) while being over 25 times faster. We note that there exists a trade off between precision and recall in QuickUMLS with respect to the similarity threshold: larger values of α increase precision but decrease recall; the opposite holds true for smaller values of α. For both the i2b2 dataset and the THYME corpus, $\alpha = 0.7$ yields to the best performances in terms of F-1 (difference is not statistically significant for the i2b2 dataset); however, in other applications, values of α that favors precision over recall (or vice versa) might be more desirable.

Lastly, the results for the drug reviews dataset are shown in Table 3. First, we notice that QuickUMLS outperforms both cTAKES and MetaMap in all metrics (precision, recall, F-1, and running time; statistically significant). We believe that QuickUMLS is more suited to handle lexical variation typical of laypeople-generated content due to the use of approximate dictionary matching algorithm. However, we notice that all systems achieve lower performances than on the other datasets. We observe that the decrease in precision is likely caused by the fact that only the symptoms and findings associated with adverse reactions to five cancer drugs have been annotated in the dataset. That means that correctly identified medical symptoms can cause a false positive if they are not an adverse drug reaction for any of the five drugs in the dataset. Finally, we attribute the decrease in recall to the fact that laypeople are more likely to describe symptoms using locutions or informal terms.

4. CONCLUSIONS

In this work we introduced QuickUMLS, a system that extract medical concepts from unstructured text. Given a document, QuickUMLS extract spans of the documents that have an approximate match in the set of strings in UMLS, returning the concepts associated with such strings. We showed that QuickUMLS offers comparable performance to state-of-the-art systems while being substantially faster.

5. ACKNOWLEDGMENTS

This work was partially supported by the US National Science Foundation through grant CNS-1204347.

6. REFERENCES

[1] A. B. Abacha and P. Zweigenbaum. Medical entity recognition: A comparison of semantic and statistical methods. In *BioNLP Workshop*, 2011.

[2] A. R. Aronson and F. M. Lang. An overview of MetaMap: historical perspective and recent advances. *JAMIA*, 17(3), 2010.

[3] S. Bethard, L. Derczynski, G. Savova, J. Pustejovsky, and M. Verhagen. SemEval 2015 task 6: Clinical TempEval. In *SemEval Workshop*, 2015.

[4] T. Delbecque and P. Zweigenbaum. Metacode: a lightweight umls mapping tool. In *AIME*, 2007.

[5] G. Divita, Q. T. Zeng, A. V. Gundlapalli, S. Duvall, J. Nebeker, and M. H. Samore. Sophia: a expedient UMLS concept extraction annotator. In *AMIA*, volume 2014, 2014.

[6] L. Goeuriot, L. Kelly, H. Suominen, L. Hanlen, A. Névéol, C. Grouin, J. Palotti, and G. Zuccon. Overview of the CLEF eHealth evaluation lab 2015. In *CLEF*. 2015.

[7] D. A. Grossman and O. Frieder. *Information Retrieval: Algorithms and Heuristics*. Springer, 2012.

[8] N. Limsopatham, C. Macdonald, and I. Ounis. Inferring conceptual relationships to improve medical records search. OAIR, 2013.

[9] N. Okazaki and J. Tsujii. Simple and efficient algorithm for approximate dictionary matching. In *Coling*, 2010.

[10] K. Roberts, M. Simpson, D. Demner-Fushman, E. Voorhees, and W. Hersh. State-of-the-art in biomedical literature retrieval for clinical cases: a survey of the TREC 2014 CDS track. *Inf. Retr. Journal*, 19(1), 2016.

[11] S. Sarawagi and A. Kirpal. Efficient set joins on similarity predicates. In *SIGMOD*, 2004.

[12] G. K. Savova, J. J. Masanz, P. V. Ogren, J. Zheng, S. Sohn, K. C. Kipper-Schuler, and C. G. Chute. Mayo clinical text analysis and knowledge extraction system (cTAKES): architecture, component evaluation and applications. *JAMIA*, 17(5), 2010.

[13] W. F. Styler IV, S. Bethard, S. Finan, M. Palmer, S. Pradhan, P. C. de Groen, B. Erickson, T. Miller, C. Lin, G. Savova, et al. Temporal annotation in the clinical domain. *Trans. of the ACL*, 2, 2014.

[14] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weissenborn, A. Krithara, S. Petridis, D. Polychronopoulos, et al. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinf.*, 16(1), 2015.

[15] Ö. Uzuner, B. R. South, S. Shen, and S. L. DuVall. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *JAMIA*, 18(5), 2011.

[16] A. Yates and N. Goharian. ADRTrace: detecting expected and unexpected adverse drug reactions from user reviews on social media sites. In *ECIR*. Springer, 2013.

[17] X. Zhou, X. Zhang, and X. Hu. MaxMatcher: Biological concept extraction using approximate dictionary lookup. In *PRICAI*, 2006.