A Learning-Based Approach to Combine Medical Annotation Results
(Short Paper)

Victor Christen[1(✉)], Ying-Chi Lin[1], Anika Groß[1], Silvio Domingos Cardoso[2,3], Cédric Pruski[2], Marcos Da Silveira[2], and Erhard Rahm[1]

[1] University of Leipzig, Leipzig, Germany
{christen,lin,gross,rahm}@informatik.uni-leipzig.de
[2] LIST, Luxembourg Institute of Science and Technology,
Esch-sur-Alzette, Luxembourg
{silvio.cardoso,cedric.pruski,marcos.dasilveira}@list.lu
[3] LRI, University of Paris-Sud XI, Gif-sur-Yvette, France

Abstract. There exist many tools to annotate mentions of medical entities in documents with concepts from biomedical ontologies. To improve the overall quality of the annotation process, we propose the use of machine learning to combine the results of different annotation tools. We comparatively evaluate the results of the machine-learning based approach with the results of the single tools and a simpler set-based result combination.

Keywords: Biomedical annotation · Annotation tool
Machine learning

1 Introduction

The annotation of entities with concepts from standardized terminologies and ontologies is of high importance in the life sciences to enhance semantic interoperability and data analysis. For instance, exchanging and analyzing the results from different clinical trials can lead to new insights for diagnosis or treatment of diseases. In the healthcare sector there is an increasing number of documents such as electronic health records (EHRs), case report forms (CRFs) and scientific publications, for which a semantic annotation is helpful to achieve an improved retrieval of relevant observations and findings [1,2].

Unfortunately, most medical documents are not yet annotated, e.g., as reported in [9] for CRFs, despite the existence of several tools to semi-automatically determine annotations. This is because annotating medical documents is highly challenging since documents may contain mentions of numerous medical entities that are described in typically large ontologies such as the Unified Medical Language System (UMLS) Metathesaurus. The mentions may also be ambiguous and incomplete and thus difficult to find within the ontologies. The

S. Auer and M.-E. Vidal (Eds.): DILS 2018, LNBI 11371, pp. 135–143, 2019.
https://doi.org/10.1007/978-3-030-06016-9_13

tools thus typically can find only a fraction of correct annotations and may also propose wrong annotations. Furthermore, the tools typically come with many configuration parameters making it difficult to use them in the best way.

Given the limitations of individual tools it is promising to apply several tools and to combine their results to improve overall annotation quality. In our previous work [11], we investigated already simple approaches to combine the results of three annotation tools based on set operations such as union, intersection and majority consensus. In this short paper, we propose and evaluate a machine learning (ML) approach for combining several annotation results.

Specifically, we make the following contributions:

- We propose a ML approach for combining the results of different annotation tools in order to improve overall annotation quality. It utilizes training data in the form of a so-called annotation vectors summarizing the scores of the considered tools for selected annotation candidates.
- We evaluate the new approach with different parameter and training settings and compare it with the results of single tools and the previously proposed combinations using set operations.

We first discuss related work on finding annotations and combining different annotation results. In Sect. 3, we propose the ML-based method. We then describe the evaluation methodology and analyze the results in Sect. 4. Finally, we conclude.

2 Related Work

Many annotation tools utilize a dictionary to store the concepts of the ontologies of interest (e.g., UMLS) to speedup the search for the most similar concepts for certain words of a document to annotate. Such dictionary-based tools include MetaMap, NCBO Annotator [8], IndexFinder [15], ConceptMapper [13], NOBLE Coder [14] cTAKES [12] and our own AnnoMap approach [7] that combines several string similarities and applies a post-processing to select the most promising annotations. There have also been annotation approaches using machine learning [4]. They can achieve good results but incur a substantial effort to provide suitable training data.

In our previous work [11], we combined annotation results for CRFs determined by the tools MetaMap, cTAKES and AnnoMap using the set-based approaches *union*, *intersection* and *majority*. The *union* approach includes the annotations from any tool to improve recall while *intersection* only preserves annotations found by all tools for improved precision. The *majority* approach includes the annotations found by a majority of tools, e.g., by at least two of three tools. Overall the set-based approach could significantly improve annotation quality, in particular for *intersection* and *majority*.

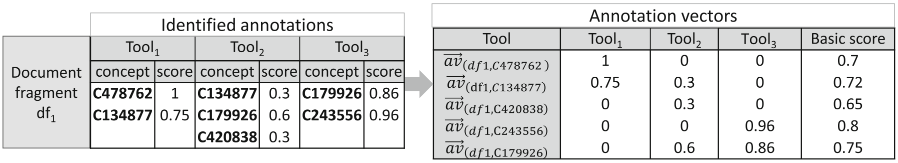
Though ML approaches have been used for annotating entities, so far they have rarely been applied for combining annotation results as we propose in this paper. Campos et al. utilized Conditional Random Fields model to recognize named entities of gene/protein terms using the results from three dictionary-based systems and one machine learning approach [5]. The learned combination could outperform combinations based on *union* or *intersection*. Our ML-based combination approach is inspired by methods proposed in record-linkage domain where the goal is to identify record pairs representing the same real-world entity [10]. Instead of a manually configured combination of different similarity values for different record attributes the ML approaches learn a classification model (e.g., using decision tree or SVM learning) based on a training set of matches and non-matches. The learned models automatically combine the individual similarities to derive at a match or non-match decision for every pair of records.

3 Machine Learning-Based Combination Approach

The task of *annotation* has as input a set of documents $D = \{d_1, d_2, \ldots, d_n\}$ to annotate, e.g., EHRs, CRFs or publications, as well as the ontology ON from which the concepts for annotation are to be found. The goal is to determine for each document fragment df (e.g., sentences) the set of its most precisely describing ontology concepts. The annotation result includes all associations between a document fragment df_j and its annotating concepts from ON. The problem we address is the *combination of multiple annotation results* for documents D and ontology ON that are determined by different tools. The tool-specific annotation results are obtained with a specific parameter configuration selected from a typically large number of possible parameter settings. The goal is to utilize complementary knowledge represented in the different input results to improve the overall annotation result, i.e., to find more correct annotations (better recall) and to reduce the number of wrongly proposed annotations (better precision).

The main idea of the proposed ML-based method is to train a classification model that determines whether an annotation candidate (df_j, c) between a document fragment df_j and a possibly annotating concept c is correct or not. The classification model is learned based on a set of positive and negative annotation examples for each tool (configuration). For each training example (df_j, c) we maintain a so-called annotation vector $\overrightarrow{av}$ with $n+1$ elements, namely a quality score for each of the n annotation tools plus a so-called *basic score*. The basic score is a similarity between df_j and c that is independently computed from the annotation tools, e.g., based on a common string similarity function such as soft-TF/IDF or q-gram similarity. The use of the basic similarity is motivated by the observation that many concepts may be determined by only one or few tools leading to sparsely filled annotation vectors and thus little input for training the classification model. The learned classification model receives as input annotation vectors of candidate annotations and determines a decision whether the annotation is considered correct or not.

Figure 1 shows sample annotation vectors for three tools and the annotation of document fragment df_1. The table on the left shows the annotations found by the tools together with their scores (normalized to a value between 0 and 1). In total, the tools identify five different concepts resulting into the five annotation



	Identified annotations					
	Tool_1		Tool_2		Tool_3	
Document fragment df_1	concept	score	concept	score	concept	score
	C478762	1	**C134877**	0.3	**C179926**	0.86
	C134877	0.75	**C179926**	0.6	**C243556**	0.96
			C420838	0.3		

Annotation vectors				
Tool	Tool_1	Tool_2	Tool_3	Basic score
$\overrightarrow{av}_{(df1,C478762\,)}$	1	0	0	0.7
$\overrightarrow{av}_{(\text{df1},C134877)}$	0.75	0.3	0	0.72
$\overrightarrow{av}_{(df1,\text{C420838})}$	0	0.3	0	0.65
$\overrightarrow{av}_{(df1,\text{C243556})}$	0	0	0.96	0.8
$\overrightarrow{av}_{(df1,\text{C179926})}$	0	0.6	0.86	0.75

Fig. 1. Sample annotations and corresponding annotation vectors

vectors shown on the right of Fig. 1. For example, the annotation of df_1 with concept C478762 has the annotation vector $\overrightarrow{av}_{(df1,C478762)}$ of $(1, 0, 0, 0.7)$ since tool 1 identified this annotation with a score of 1, tools 2 and 3 did not determine this annotation (indicated by score 0), and the basic score is 0.7.

We use three classifiers: decision tree, random forest and support vector machines (SVM), to train classification models. A *decision tree* consists of nodes and each node represents a binary decision function based on a score threshold of a tool, e.g. $score_{\text{MetaMap}} > 0.7$. When an annotation vector $\overrightarrow{av}$ is input into a decision tree, decisions are made from the root node to the leaf node according to the values of $\overrightarrow{av}$. As output, $\overrightarrow{av}$ is classified as a correct or incorrect annotation. Random forest [3] utilizes an ensemble of decision trees and derives the classification decision from the most voted class of the individual decision trees. To determine a random forest classification model, each decision tree is trained by different samples of the training dataset. The goal of an *SVM* is to compute a hyperplane that separates the correct annotation vectors (represents a true annotation) from the incorrect ones. To separate vectors that are not linearly separable, SVM utilizes a kernel function to map the original vectors to a higher dimension so that the vectors can be separated.

A key step for the ML-based combination approach is the provision of suitable training data of a certain size. For this purpose, we determine annotation results with different tools and a specific configuration for a set of training documents. From the results we randomly select a subset of n annotations and generate the corresponding annotation vectors AV_{train} and label them as either correct or incorrect annotations. Providing a sufficient number of positive and negative training examples is of high importance to determine a classification model with enough discriminative power to correctly classify annotation candidates. To control the ratio between these two kinds of annotations we follow the approach of [10] and use a parameter *tpRatio* (true positive ratio). For instance, $tpRatio = 0.4$ means 40% of all annotations in AV_{train} are correct. In our evaluation, we will consider the influence of both the training size n and *tpRatio*.

4 Evaluation and Results

We now evaluate our ML-based combination approach and compare it with the simpler set-based combination of annotation results. After the description of the experimental setup we analyze the influence of different training configurations and learners. In Sect. 4.3, we compare the results of the ML approach with the single tools and set-based combination. The evaluation focuses on the standard metrics *recall*, *precision* and their harmonic mean *F-measure* as the main indicator for annotation quality.

4.1 Experimental Setup

We use two datasets with medical forms (CRFs) for which a reference mapping exists: a dataset with forms on *eligibility criteria* (EC) and a dataset with *quality assurance* (QA) forms. The EC dataset contains 25 forms with 310 manually annotated questions. The QA dataset has 24 standardized forms with 543 annotated questions used in cardio-vascular procedures. The number of annotations in the reference mappings is 541 for EC and 589 for QA. These datasets have also been used in previous annotation evaluations [6,7] and turned out to be very challenging. For annotation we use five UMLS ontologies of version 2014AB: UMLS Metathesaurus, NCI Thesaurus, MedDRA, OAC-CHV, and SNOMED-CT_US. Since we use different subsets of UMLS in this paper and in the previous studies [7], the results are not directly comparable.

As in our previous study [11] we combine annotation results of the tools MetaMap, cTAKES and AnnoMap and apply the same set of configurations. In the annotation vectors, we use the normalized scores of the tools and determine the *basic score* by using soft-TF/IDF. For the classifiers (decision tree, random forest, SVM) we apply Weka as machine learning library. We generate training data of sizes 50, 100 or 200 selected from the *union* of the three tools. A *tpRatio* $\in \{0.2, 0.3, 0.4, 0.5\}$ is applied for each sample generation. For each ML test configuration (i.e., choice of classifier, sample size, *tpRatio* and tool configuration) we produce three randomly selected training sets and use each to generate a classifier model so that our results are not biased by just one sample. For each test configuration we measure average precision, average recall and macro F-measure that is based on the average precision and the average recall.

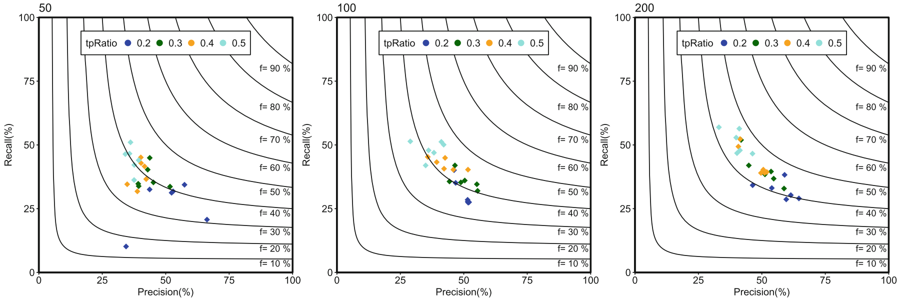


Fig. 2. Precision/recall results for different *tpRatio* values and training sizes n (dataset EC, random forest learning)

4.2 Machine Learning-Based Combination of Annotation Tools

For the analysis of our ML-based combination approach we first focus on the impact of parameter *tpRatio* and the size of the training sets. We then compare the three classifiers decision tree, random forest and SVM. Due to space restrictions we present only a representative subset of the results.

Figure 2 shows the annotation quality for dataset EC using random forest learning for different *tpRatios* (0.2 to 0.5) and three different training sizes (50, 100 and 200). Each data point represents the classification quality according to a certain *tpRatio* with a certain configuration of the considered tools. We observe that data points with the same *tpRatios* are mostly grouped together indicating that this parameter is more significant than other configuration details. We further observe for all training sizes that models trained with a larger *tpRatios* of 0.5 or 0.4 tend to reach a higher recall (but lower precision) than for smaller *tpRatios* values. Apparently low *tpRatio* values provide too few correct annotations so that the learned models are not sufficiently able to classify correct annotations as correct. By contrast, higher *tpRatio* values can lead to models that classify more incorrect annotations as a correct thereby reducing precision. For random forest, a *tpRatio* of 0.4 is generally a good compromise setting.

Figure 2 also shows that larger training sizes tend to improve F-measure since the data points for the right-most figure (training size $n = 200$) are mostly above the F-measure line of 50% while this is not the case for the left-most figure ($n = 50$). Figure 3 reveals the influence of the training size in more detail by showing the macro-average precision, recall and F-measure obtained by random forest using different training sizes. For both datasets, EC and QA, we observe that larger training sizes help to improve both precision and recall and thus F-measure. Hence, average F-measure improved from 40.1% to 42.5% for dataset EC and even from 52.0% to 56.9% for QA when the training size increases from 50 to 200 annotation samples.

Figure 4 depicts the macro-average precision, recall and F-measure over different *tpRatios*, sample sizes and configurations. For both datasets, random forest obtains the best recall values (EC: 40.0%, QA: 46.8%) while decision tree

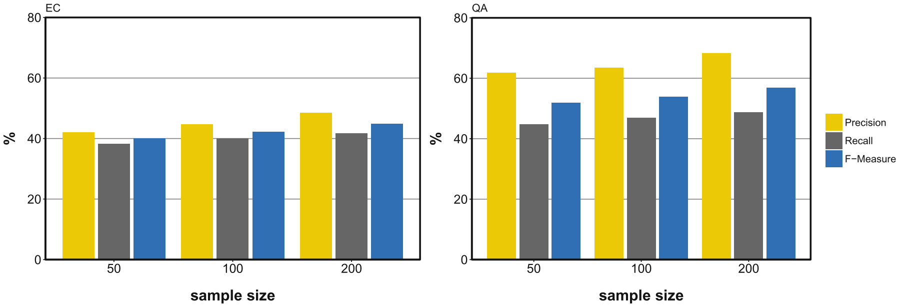


Fig. 3. Impact of training sizes on annotation quality for datasets EC and QA

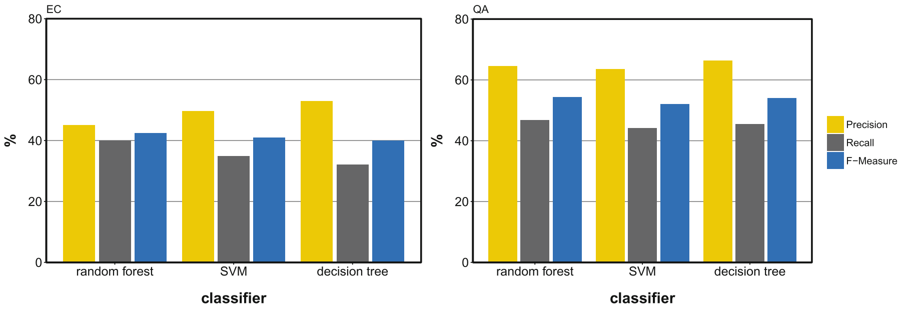


Fig. 4. Average annotation quality for random forest, SVM and decision tree.

achieves the best precision (EC: 52.9%, QA: 66.4%). In terms of average F-measure the three learning approaches are relatively close together, although random forest (42.4%) outperforms SVM and decision tree by 1.4% resp. 2.5% for EC. For the QA dataset, random forest (54.3%) outperforms decision tree and SVM by 0.3% resp. 2.2%. Moreover, we experimentally tested our approach with or without using the basic scores in addition to the tool results. We observed that using the basic score improves F-Measure by 1.6% (EC) and 1% (QA), indicating that it is valuable to improve annotation results.

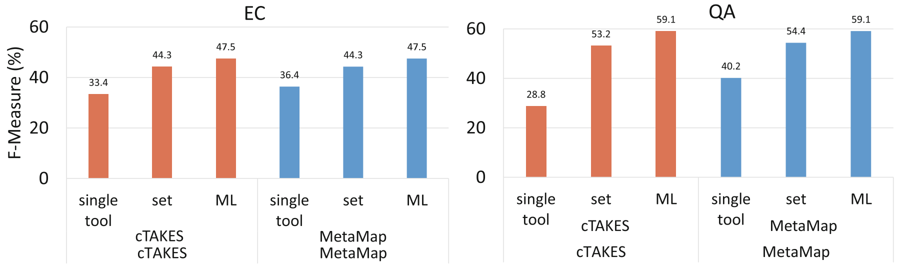


Fig. 5. Summarizing F-measure results for cTAKES and MetaMap and the set-based and ML-based result combinations for the EC and QA datasets.

4.3 Comparison with Set-Based Combination Approaches

We finally compare the annotation quality for the ML-based combinations with that of the individual tools cTAKES and MetaMap as well as with the results for the set-based combinations proposed in [11]. Figure 5 summarizes the best F-measure results for both datasets. We observe that the F-measure of the individual tools is substantially improved by both the set-based and ML-based combination approaches, especially for cTAKES (by about a factor 3–4.5). The ML-based combination outperforms the set-based combinations for both datasets. Consequently, the best results can be improved for EC (from 44.3% to 47.5%)

and QA (from 56.1% to 59.1%) by using a sample size of 200. This underlines the effectiveness of the proposed ML-based combination approach.

5 Conclusions

The annotation of documents in healthcare such as medical forms or EHRs with ontology concepts is of high benefit but challenging. We proposed and evaluated a machine learning approach to combine the annotation results of several tools. Our evaluation showed that the ML-based approach can dramatically improve the annotation quality of individual tools and that it also outperforms simpler set-based combination approaches. The evaluation showed that the improvements are already possible for small training sizes (50–200 positive and negative annotation examples) and that random forest performs slightly better than decision tree or SVM learning. In future work, we plan to apply the ML-based combination strategy to annotate further kinds of documents and to use machine learning also in the generation of annotation candidates.

References

1. TIES-Text Information Extraction System (2017). http://ties.dbmi.pitt.edu/
2. Abedi, V., Zand, R., Yeasin, M., Faisal, F.E.: An automated framework for hypotheses generation using literature. BioData Min. **5**(1), 13 (2012)
3. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
4. Campos, D., Matos, S., Oliveira, J.: Current methodologies for biomedical named entity recognition. In: Biological Knowledge Discovery Handbook: Preprocessing, Mining, and Postprocessing of Biological Data, pp. 839–868 (2013)
5. Campos, D., et al.: Harmonization of gene/protein annotations: towards a gold standard MEDLINE. Bioinformatics **28**(9), 1253–1261 (2012)
6. Christen, V., Groß, A., Rahm, E.: A reuse-based annotation approach for medical documents. In: Groth, P., et al. (eds.) ISWC 2016. LNCS, vol. 9981, pp. 135–150. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46523-4_9
7. Christen, V., Groß, A., Varghese, J., Dugas, M., Rahm, E.: Annotating medical forms using UMLS. In: Ashish, N., Ambite, J.-L. (eds.) DILS 2015. LNCS, vol. 9162, pp. 55–69. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21843-4_5
8. Dai, M., et al.: An efficient solution for mapping free text to ontology terms. In: AMIA Summit on Translational Bioinformatics, vol. 21 (2008)
9. Dugas, M., et al.: Portal of medical data models: information infrastructure for medical research and healthcare. Database: J. Biol. Databases Curation (2016)
10. Köpcke, H., Thor, A., Rahm, E.: Learning-based approaches for matching web data entities. IEEE Internet Comput. **14**(4), 23–31 (2010)
11. Lin, Y.-C., et al.: Evaluating and improving annotation tools for medical forms. In: Da Silveira, M., Pruski, C., Schneider, R. (eds.) DILS 2017. LNCS, vol. 10649, pp. 1–16. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69751-2_1
12. Savova, G.K., et al.: Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. JAMIA **17**(5), 507–513 (2010)

13. Tanenblatt, M.A., Coden, A., Sominsky, I.L.: The ConceptMapper approach to named entity recognition. In: Proceedings of LREC, pp. 546–551 (2010)
14. Tseytlin, E., Mitchell, K., Legowski, E., Corrigan, J., Chavan, G., Jacobson, R.S.: NOBLE-Flexible concept recognition for large-scale biomedical natural language processing. BMC Bioinform. **17**(1), 32 (2016)
15. Zou, Q., et al.: IndexFinder: a knowledge-based method for indexing clinical texts. In: Proceedings of AMIA Annual Symposium, pp. 763–767 (2003)