NLP project 2 - final report
SLR - NieLeniweProjekty

First Author: Michał Gozdera
Warsaw University of Technology
01142172@pw.edu.pl

Second Author: Małgorzata Hadasz
Warsaw University of Technology
01156169@pw.edu.pl

Third Author: Krystian Kurek
Warsaw University of Technology
01121582@pw.edu.pl

supervisor: Anna Wróblewska
Warsaw University of Technology
anna.wroblewska1@pw.edu.pl

Abstract

Nowadays, the rapid increase in knowledge and the amount of performed research in numerous domains (like Computer Science and Medicine) causes the need for solutions designed to segregate, organise and find created papers and publications automatically. A key aspect of such frameworks is to detect the topic of a given article and connect the discovered subject with domain-specific concepts. This work shows an extension to Project 1 created on Natural Language Processing classes at Warsaw University of Technology in the winter semester 2022/2023. Project 1 focused on automating a part of Semantic Literature Reviews, namely extracting keywords from scientific papers, tagging them with domain ontologies and performing disambiguation. Project 2 extends Project 1 with several improvements. Firstly, we enhance the evaluation process, as Project 1 turned out to be a non-trivial and complex task. Then, we implement several enhancements in the disambiguation process. Finally, we also examine the influence of system parameters on its performance.

zmienic tableke

Work assigment	Person
BERTopic for CRAFT	M. Gozdera (6.66%, 6h)
LDA & BERTopic hyperparams tuning	M. Gozdera (8.88%, 8h)
Keywords extraction evaluation	M. Gozdera (17.79%, 16h)
UMLS extraction	K. Kurek (11.11%, 24h)
Embedding tagger	K. Kurek (11.11%, 10h)
Statistics summary	K. Kurek (11.11%, 14h)
LDA adjustment	M. Hadasz (2.2%, 2h)
Data analysis	M. Hadasz (18.7%, 17h)
Disambiguation improvements	M. Hadasz (12.1%, 11h)

Table 1: Table of contents

1 Introduction

Our goal in Project 2 is to improve the evaluation process of the current solution. As it turned out in the outcomes of the first project, evaluation of the task we approached is not trivial, so in Project 2, we focus on enhancing it.

In the rest of this section, we describe the significance of the project, project activities, timeline, specific research goals, and post-doc risk analysis. Section 2 is a brief summary of what we did in Project 1. Section 3 describes related works (state-of-the-art and data used). In sections 4 and 5 we present the research approach we incorporated, the methods we implemented and their results. Section 5 includes the core part of Project 2, which is the evaluation analysis, hyperparameters influence and different disambiguation variants. We conclude and propose directions for future work in Section 6.

1.1 Significance of the project

The evaluation phase of each AI algorithm is essential in order to validate it and ensure it works properly. In the first project, we created a solution that defines an end-to-end process of extracting semantic keywords for systematic literature reviews. Based on our research, there exist models and algorithms that can be successfully modified and combined to create a well-performing method tackling the scientific problem we describe (Jonquet et al., 2009; Lee et al., 2019; Blei et al., 2003; Grootendorst, 2022b).

The second project's significance is self-explainable. Solving a problem of SLR automa-

Date	Stage name	Description
9.12.2022	Project proposal	literature review, solution concept and proposal
5.01.2023	Proof of concept	data preparation, methods adjustment, new disambiguation options
20.01.2023	Final project	full solution and prepared product

Table 2: Project activity and timeline

tion validation is not trivial and, at the same time, essential to create solutions applicable in real-life scenarios.

1.2 Project activities and timeline

We divided our project into 3 main parts, presented in Table 2.

1.3 Specific research goals

We established the following general tasks and research goals for the project:

- investigating the performance of proposed solution on the CRAFT (Cohen et al., 2017) data set,
- comparison of the proposed method performance between CRAFT (Cohen et al., 2017) and MedMentions datasets (Mohan and Li, 2019),
- evaluation for the entire pipeline (keywords extraction, tagging, disambiguation) on the CRAFT dataset,
- creating an evaluation technique for the keywords extraction step (instead of evaluating it only as a part of the whole pipeline),
- investigating the influence of hyperparameters in the keywords extraction process,
- disambiguation improvement – initial sorting, weighted voting, forced decisions approaches (details in section 5.4),
- incorporating a new tagger - MedCAT (Kraljevic et al., 2021).

Moreover, in Project 2 we adjusted details in all stages of the project (f.e., the number of topics in the LDA algorithm).

1.4 Risk analysis

After working on and developing the project, we want to summarize the risks we predicted and experienced. Concerning the first two identified risks in privacy policies of algorithms or datasets, we did not experience any of those threads. For both data and algorithms, the privacy policies did not change, and therefore we were able to use them. Also, with regard to the risk to the team, we did not experience any problems with teamwork. All the members worked equally and with engagement and did not make any serious mistakes.

2 Project 1 - recap

In Project 1 our aim was to address the question of finding semantic keywords for a systematic literature overview. Namely, we proposed different solutions to extract keywords from medical papers abstracts, tagged these keywords with ontologies concepts and chose the best tags based on the disambiguation technique.

For keywords extraction, we focused mainly on the BERTopic (Grootendorst, 2022b) model, but we also compared it with the LDA (Blei et al., 2003) method.

One approach to keyword tagging was the use of NBCO annotator (Jonquet et al., 2009) (standard and simple solution). At the same time, the other was implemented from scratch with the use of the word embedding concept (*bioBERT* (Lee et al., 2019)).

Tag disambiguation techniques relied on the Closest Sense method (Alexopoulou et al., 2009), adjusted to our problem statement.

We decided to implement the solution described above, because, to the best of our knowledge, there is currently no state-of-the-art solution combining those three functionalities and taking advantage of the latest NLP solutions. Authors of (van Dinter et al., 2021) mention the 12 steps that are defined in Systematic Literature Overview (SLR) domain. They are divided into three main categories, which are: Need for a review, Conducting the review, and Reporting the review. They claim to find 41 studies approaching automating one or more selected steps in the SLR process, mainly for the Software Engineering and Medical domain. Our project fo-

cused on Conducting the review. We tried to automate steps: Identification of research, Selection of primary studies, Study quality assessment by keywords extraction and ontology tagging. What differed our approach from the ones currently available is the use of BERTopic to extract keywords from papers. What is more, the combination of keywords extraction, ontology tagging and disambiguation was not covered in the literature to the best of our knowledge. We show the workflow of Project 1 in Figure 1.

3 Related works

3.1 Current solutions and state-of-the-art

Currently, available solutions are not specifically directed toward the aim we presented. There exist state-of-the-art solutions performing specific parts of what we were going to implement, but the entire process itself is not well investigated, in our opinion. As mentioned in Sections 1 and 2, we focused on automating Conducting the review part of Semantic Literature Reviews (van Dinter et al., 2021).

Regarding keywords extraction, Latent Dirichlet Allocation - LDA (Blei et al., 2003) is one of the state-of-the-art algorithms. However, it is usually replaced by models utilizing modern word embedding techniques, like BERTopic (Grootendorst, 2022b).

For a long time, simple solutions for ontology-based tagging in medical data, like NCBO annotator (Jonquet et al., 2009) were used. Recently, more sophisticated approaches (like ScispaCy (Neumann et al., 2019)) appeared. What is more, the word embedding idea is getting more and more interest in the NLP field, actually being the current state-of-the-art word representation (mainly because of its high performance and important properties, like preserving semantic meaning). However, to the best of our knowledge, no state-of-the-art embedding-based annotator is provided for the use of specific medical ontologies.

According to our research, ontology tag disambiguation is the least explored part of the solution. There are not many papers approaching this topic, most of them treating disambiguation as a side part of other solutions (Leaman and Lu, 2016), (Bindelli et al., 2008a). An algorithm that seems to fit the needs of our solution best is the Closest Sense method (Alexopoulou et al., 2009).

More detailed description of all the solutions can be found in the Appendix.

3.2 Results of preliminary research

Introductory research resulted in gathering knowledge about state-of-the-art methods than can be incorporated into our solution. They are described in Section 3. Apart from reading about particular solutions, we also tested and verified existing implementations:

- Keywords extraction - the LDA algorithm is available in *sklearn* library (Buitinck et al., 2013); BERTopic can be found in *bertopic* package (Grootendorst, 2022a),
- Keywords tagging – for NCBO annotator, the REST API is available (Jonquet et al., 2021), so our solution is based on the use of it via HTTP connection; there is no concrete implementation of embedding-based annotations with medical ontologies that would satisfy our need, so this part is implemented from scratch based on *BioBert* (Lee et al., 2019) embeddings.
- Tags disambiguation – since we are going to use a modified version of the Closest Sense method (Alexopoulou et al., 2009), we implemented it from scratch.

The main result of the preliminary research was that currently available solutions (with quite a few modifications) should allow us to develop a fully usable method for extracting semantic keywords. However, no specific algorithms pipeline (combining all the above methods) is available now. Creating one (Project 1) and performing its detailed evaluation (Project 2) was the goal of our projects.

3.3 Data

Taking into account the disadvantages of the MedMentions (Mohan and Li, 2019) dataset mentioned in Section 2, which came after the first project, we decided to test our solution on a different set of data. We incorporated the CRAFT dataset (Cohen et al., 2017). It contains 97 publications (which is far less than MedMentions (Mohan and Li, 2019) containing over 4000 publications). CRAFT dataset is annotated with these ontologies:

- CHEBI - Chemical Entities of Biological Interest(Hastings et al., 2015),

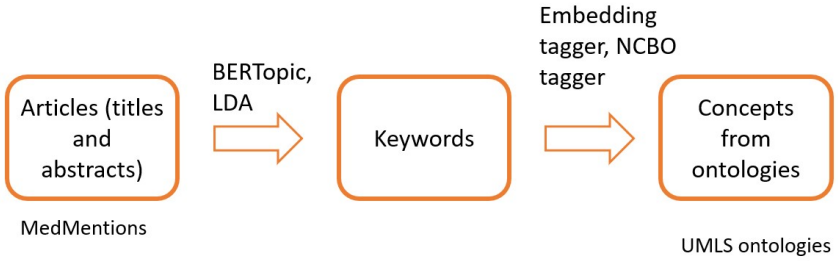

Figure 1: Workflow of the Project 1.

- CL - Cell Ontology(Diehl et al., 2016),
- GO - Gene Ontology(Ashburner et al., 2000),
- MOP - Molecular Process Ontology (MOP,),
- NCBITaxon - NCBI Taxonomy (Schoch et al., 2020),
- PR - Protein Ontology (Natale et al., 2016),
- SO - Sequence Ontology (Eilbeck et al., 2005),
- UBERON - Uberon (Haendel et al., 2014).

In order to reuse the solutions, we implemented in the first project, the dataset was transformed into a format matching the one used before. Details of those transformations are presented in section 3.3.1.

3.3.1 Data preprocessing

To prepare data for the topic extraction a few preprocessing steps needed to be performed. The preprocessing step was also preceded by in-depth data analysis.

As mentioned in Section 3.3 the dataset contains 97 articles, which are given in the .txt format. We wanted to transform them into the format we used for the MedMentions (Mohan and Li, 2019) dataset. To achieve that we needed to divide the articles into titles and the rest. It was a trivial task since the title was always only the first line. We also wanted to extract just abstracts from each article. This task was more challenging because there were a few different articles format. The first one contained the title *"Abstract"* and the text after it being the abstract content (up to the next title f.e. *"Introduction"*). The second one contained the abstract divided into subsections. The last one was a single document with a different format. The first tests showed that using only abstract leads to poor results. Therefore we decided to work with the whole text. Since the dataset is small the memory and time limitations were not an obstacle.

After this data exploration and the title content division, the preprocessing was performed. Firstly, we divided each text into sentences (using the *sent_tokenize* function from the nltk library (Bird et al., 2009)). We tokenized each sentence into words (using the *word_tokenize* function from the nltk library (Bird et al., 2009)) and removed the stop words. Subsequently, we perform stemming (using *PorterStemmer* from nltk library (Bird et al., 2009)) and lemmatization (using *WordNetLemmatizer* from nltk library (Bird et al., 2009)). As a result, we got 5 new dataframe columns: *tokenized_sentences*, *tokenized_words*, *tokenized_words_no_stopwords*, *tokenized_words_processed* and *tokenized_words_lemmatize*.

We did not divide the dataset into training and test parts since all algorithms can be applied in a non-supervised manner. The reason and details are described in Section 5.1.

3.3.2 Exploratory data analysis

To get better insight into annotations and their differences across ontologies, we performed exploratory data analysis. In Figure 2 the number of annotations per document can be seen. It is visible, that those numbers vary across ontologies.

We also analyzed the most popular annotations for each ontology. This also significantly varied across different ontologies, as presented in Figure 3.

4 Approach & research methodology

Our approach and research methodology consist of several steps.

Firstly, we investigated what difficulties we encountered in the evaluation process of Project 1

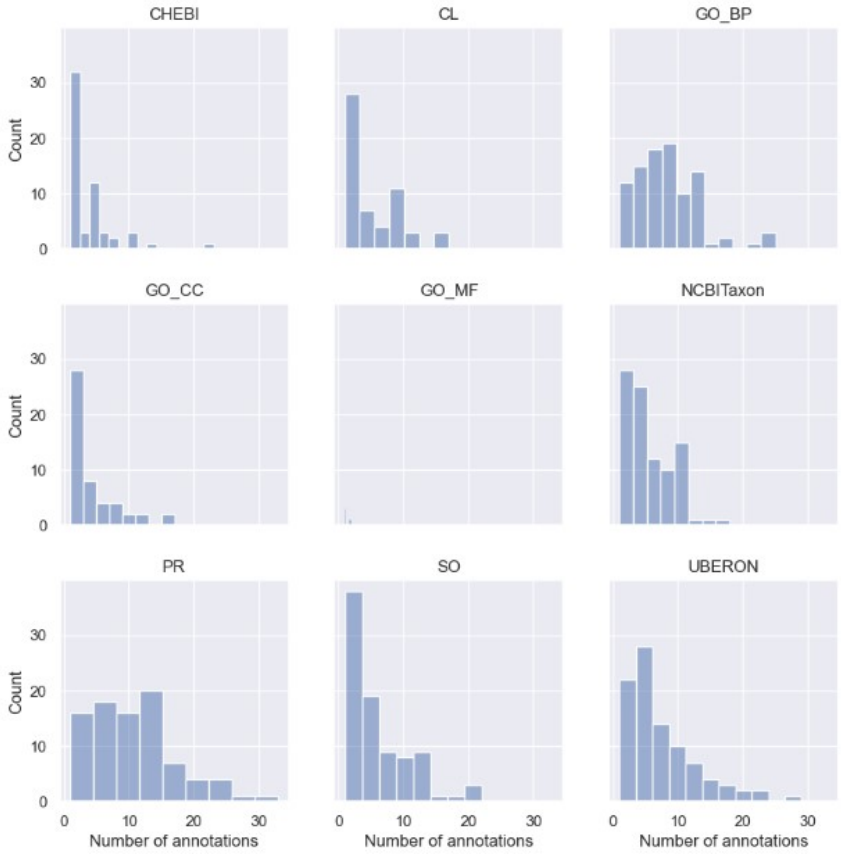

Figure 2: Number of annotations perf ontlogy

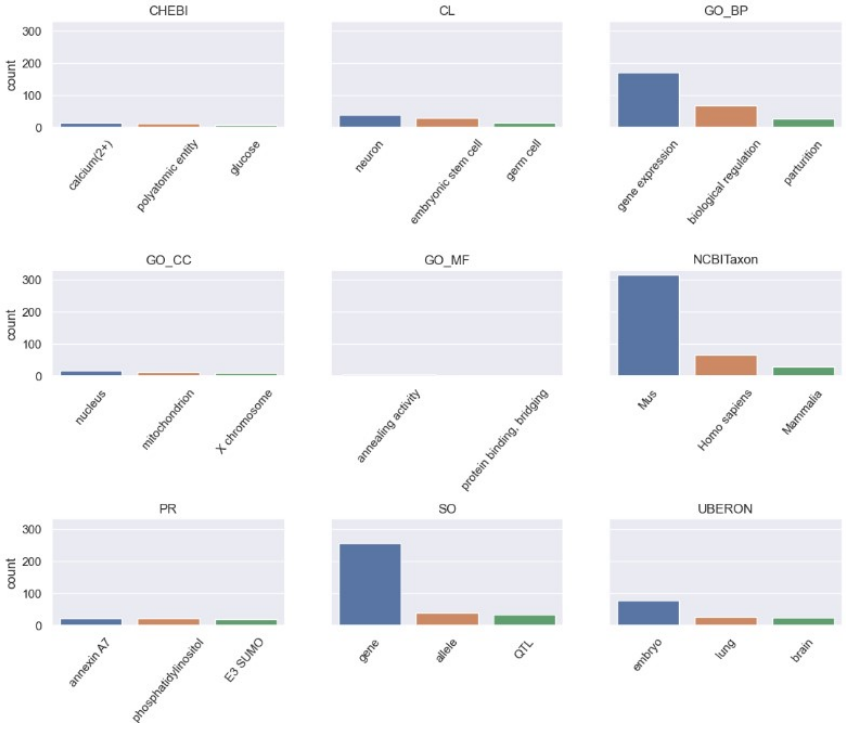

Figure 3: The most popular annotations for each ontology

and what improvements can be done. This part resulted in defining the goals for Project 2 described in Section 1.3.

Next, we prepared a Proof of Concept (PoC) so-

lution that utilised some of the concepts included in previous sections. It consisted of a new dataset and first investigation into metrics calculated on them, disambiguation improvements and plans for another evaluation methods.

Then, we prepared the final solution, including all methods described in Section 5 and the Appendix.

Each stage of the project was presented in front of other researchers working on similar projects and the project supervisor.

5 Solution analysis, result discussion and tests

In this section we show our solutions together with the evaluation methods we managed to define. Moreover, we show the evaluation results and discuss them.

5.1 Keywords extraction methods

The algorithm pipeline for CRAFT dataset (Cohen et al., 2017) is very similar to the one we used for MedMentions (Mohan and Li, 2019) in Project 1. Each method we use is described in details in the Appendix. What is more, we tuned the hyperparameters for the keywords extraction step and we run the entire pipeline for the best hyperparameters. Finally, we compared the results. Below we describe keywords extraction steps:

1. **BERTopic keywords extraction**. We changed the minimal topic size from 10 to 3 (since the number of data points is lower than in MedMentions). The algorithm managed to extract nine topics. We take ten keywords from each topic. Four first keywords from each topic are visualized in Figure 4. After analysis of hyperparameters (details in 5.2.4), we also performed BERTopic for minimal topic size equal to 6 and 22 keywords extracted from each paper.

2. **LDA keywords extraction** To adjust the number of topics to the one extracted from BERTopic we changed the number of the expected topics to 9 and kept 10 keywords per topic (default setting). After analysis of hyperparameters (details in 5.2.4), we also performed LDA for 7 topics and 22 keywords extracted from each paper.

The main difference here compared to the previous project is a data set division for the BERTopic clustering phase. Our previous approach (applicable for larger datasets like MedMentions (Mohan and Li, 2019)) was to define clusters on a train set and then, in the testing phase, assign each datapoint from the test set to a cluster. For smaller datasets, the clustering needs to be performed on the whole dataset at once (dividing CRAFT (Cohen et al., 2017) into train and test sets resulted in assigning the *outlier* topic to each datapoint from the test dataset when using the BERTopic algorithm).

	Topic	Count	Name
0	-1	24	-1_gene_protein_mouse_cell
1	0	18	0_mouse_strain_muscle_background
2	1	16	1_cell_mouse_embryonic_mutant
3	2	7	2_protein_pax6_photoreceptor_mcoln1
4	3	7	3_itpr1_axon_adam11_adam22
5	4	7	4_olfactory_receptor_mouse_dopamine
6	5	5	5_ear_sensory_hair_cell
7	6	5	6_pulmonary_individual_lung_development
8	7	4	7_sox1_annexin_a7_neuronal
9	8	4	8_bone_differentiation_slow_limb

Figure 4: Four first keywords from each topic extracted by BERTopic on CRAFT dataset.

5.2 Keywords extraction - evalutaion

As mentioned in Section 1, we aimed to develop a method to evaluate keywords extraction separately, rather than in an algorithms pipeline. Evaluating keywords extraction, ontologies tagging and disambiguation at once is very important, but at the same time, very complex. It gives the most relevant information on how the entire system works but makes it unable to detect weak parts of single pipeline steps. That is why we designed a dedicated evaluation approach for keywords extraction.

5.2.1 Metrics definition

We utilise 3 metrics allowing to evaluate keywords extraction process. They are defined as follows:

$$precision = \frac{TP}{TP + FP} \tag{1}$$

$$recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{3}$$

Where:

- True positives (TP) - number of cases when extracted keyword is (a part of) the mention from the golden standard dataset,
- False positives (FP) - number of cases when extracted keyword is not (a part of) any mention from the golden standard dataset,
- False negatives (FN) - number of cases when a mention from the golden standard dataset is not present in extracted keywords.

One can notice that these metrics are similar to the ones defined for the evaluation of the whole pipeline, used in Project 1 and recalled in Section 5.5. Here, however, we do not compare the annotations from the ground truth with annotations made by our system. Instead, we look only at ground-truth mentions and compare them with keywords extracted by BERTopic or LDA. Mentions are fragments of original texts that were tagged. The annotations here are totally skipped since this part has nothing to do with annotating. Our intuition is that people who were tagging the papers have chosen mentions that would represent *key* aspects of the papers, so they should be roughly matched.

5.2.2 Metrics results for keywords extraction

In order to have a clue about the number of keywords we should extract for each article, we checked how many mentions were defined for each paper in CRAFT dataset. Figure 5 shows the histogram of the number of mentions defined for each article.

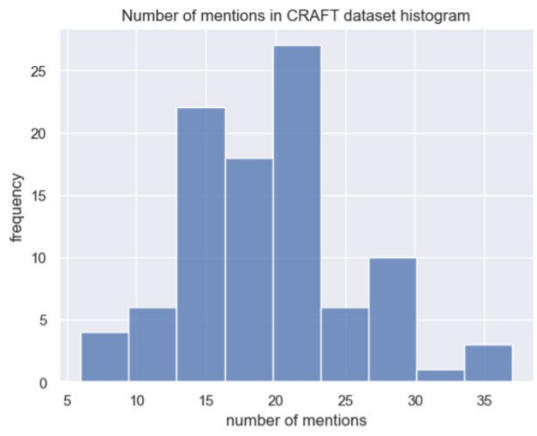

Figure 5: Histogram of the number of mentions per paper in CRAFT.

One can conclude that the range of numbers of keywords to extract can be defined as [1, 30] to roughly match the number of mentions extracted by authors of the CRAFT dataset. That is why we fit BERTopic and LDA models for a different number of keywords extracted and investigate precision, recall, and f1-score. Figure 6 depicts the obtained results.

Obviously, the precision is higher when we extract a small number of keywords because there is not a lot of False-Positives (keywords not matching any mention). Once we increase the number of keywords, more False-Positives occur and the precision goes down. However, recall increases - we have fewer and fewer False-Negatives since more real mentions are included in returned keywords.

For both LDA and BERTopic, the trends are similar. One can notice that for 22 keywords extracted, precision, recall, and f1-score achieve the same values (cross-point). This is also the point where f1-score achieves its highest value. This result is consistent with Figure 5. We obtain the highest f1-score when we extract the number of keywords roughly equal to the average number of mentions in CRAFT papers.

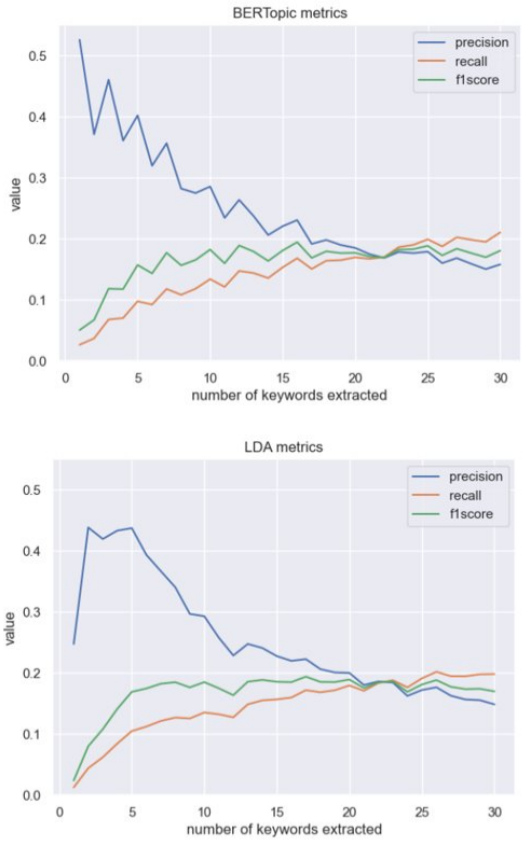

Figure 6: Precision, recall and f1-score for BERTopic and LDA keywords extraction.

5.2.3 Keywords extraction - qualitative analysis

The next part of keywords extraction is the analysis of BERTopic and LDA results based on concrete examples. We aim to understand and evaluate the tagging process. For this purpose, we investigate two outcomes for BERTopic and two for LDA: good and bad results.

Figure 7 and 8 show good and poor results obtained for different papers by BERTopics. In the former case, the True-Positive value is 10 (BERTopic manage to find 10 keywords matching ground-truth mentions), and in the latter - 0 (BERTopic did not manage to find any keyword matching ground-truth mentions).

Ground truth mentions	BERTopic keywords
polyatomic entity, germ cell, primordial germ cell, oocyte, sperm, egg cell, embryonic stem cell, embryonic cell, cell fate specification, gene expression, gastrulation, germ cell development, embryo development, fertilization, embryonic cleavage, Mus, Metazoa, Mammalia, Diptera, developmental pluripotency-associated protein 3, gene, embryo, adult organism, gonad, cell layer, 4 cell stage, cleavage stage	cell, mouse, gene, expression, embryonic, embryo, development, sex, reporter, growth, chromatin, mutant, recombination, protein, germ, heterochromatin, pten, abstract, signaling, stem, stage, loss

Figure 7: BERTopic keywords extraction for paper with PMID=15018652 (good result)

Let us remind, that BERTopic finds keywords, not phrases. That is why, by a keyword matching with mention, we understand a keyword that appears as a part of the mentioned phrase. In Figure 7 one can notice that, for example, *cell* keyword is a reasonable choice. It is part of many mentions and is informative for a given paper. Similarly (with different numbers of occurrences) for *gene*, *embryonic*, *protein*, *germ*, *stem*. All these keywords have a medical sense and are representative of the paper.

Ground truth mentions	BERTopic keywords
gene expression, parturition, death, fertilization, neurogenesis, proteolysis, extracellular matrix, membrane, Mus, Homo sapiens, Mammalia, disintegrin and metalloproteinase domain-containing protein 22, polypeptide_domain, gene, nerve, cerebral cortex, cerebellum, nervous system	itpr1, axon, adam11, adam22, ataxia, seizure, mouse, observed, disorder, deletion, result, molecule, brunol4, gjc, task, peripheral, mutation, show, human, epilepsy, homozygote, calvarial

Figure 8: BERTopic keywords extraction for paper with PMID=15018652 (bad result)

BERTopic captured also keywords that are not necessarily medical, but also important in the ground truth and are parts of mentions, like *development*, *stage*, *expression*.

Another group of keywords extracted by BERTopic are words that are not mentioned in the ground-truth CRAFT data, but they are indeed important and could be classified as mentions. The best example visible in Figure 7 is *mouse*. To prove it, in Figure 9 we include the abstract of the paper from which results are depicted in Figure 7. From the abstract, it is obvious that mice are the subject of study, so the *mouse* keyword is reasonable.

Finally, there is a group of keywords that do not match any mention and it is justified. For example, *abstract* and *loss* are not connected with the medicine domain or subject of the article. *Abstract* has been chosen as a keyword because it was probably often referenced in the paper and *loss* is a concept connected more with some algorithms/research questions than medicine itself.

In Figure 8, one can see an example of the case when BERTopic did not manage to extract any keyword being part of some ground-truth mentions. However, one can see that the majority of the keywords extracted are indeed from the medical domain, for example, *itpr1*, *adam11*, *adam22* are names of the proteins connected with genes and *mutation*, *homozygote*, *deletion* are concepts

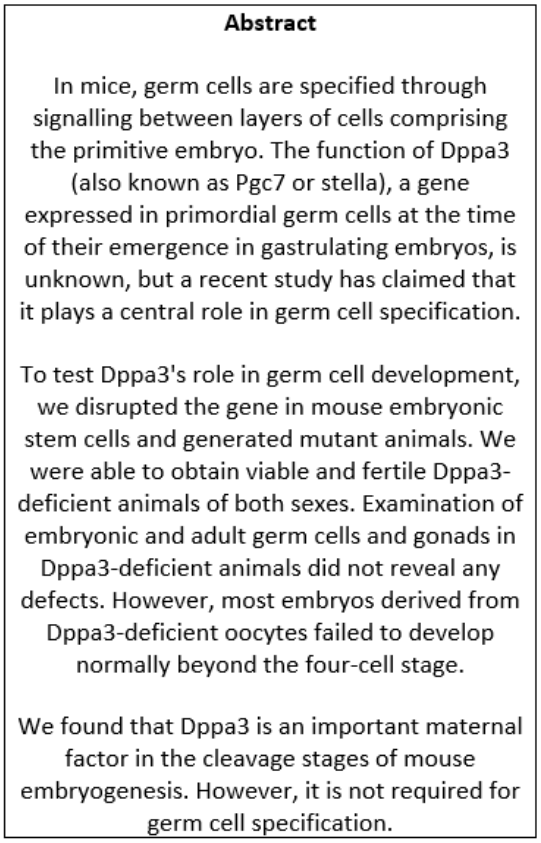

Figure 9: Abstract of paper with PMID=15018652, corresponding to results depicted in Figure 7 (good result).

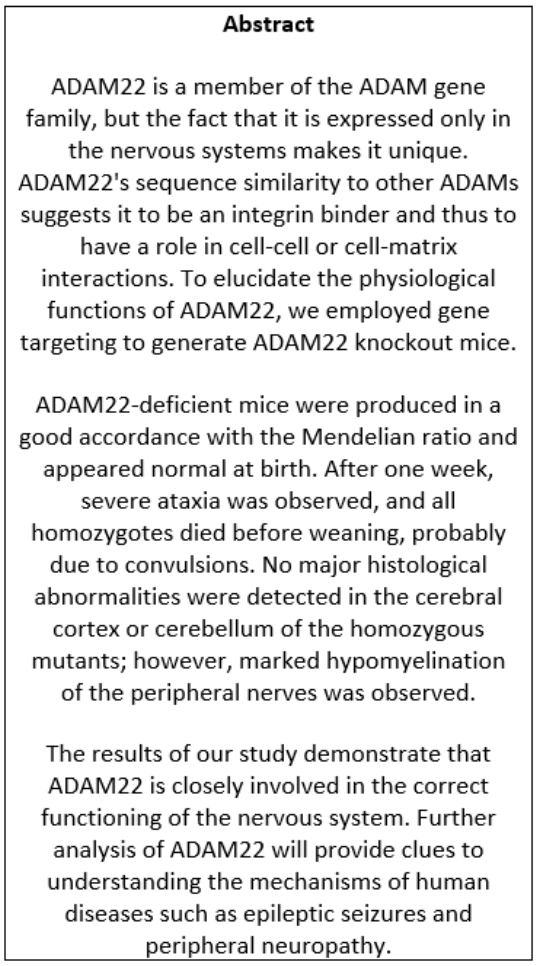

Figure 10: Abstract of paper with PMID=15876356, corresponding to results depicted in Figure 8 (bad result).

connected with genetics in general. Based only on these keywords, it can be concluded that the paper is about genetics, even though CRAFT mentions do not fit well with BERTopic keywords. To prove it, we show the abstract of the paper in Figure 10.

Analogous analyses are obtained for LDA. Our investigations showed that both LDA and BERTopic work similarly, taking into account both the calculated metrics (Figure 6) and the qualitative analysis. For convenience, we include analogous results for LDA (showing both good and bad results according to the number of True-Positives) in Figures 11 and 12.

5.2.4 Hyperparameters influence on the keywords extraction process

We evaluated how the efficiency of proposed keywords extractors changes with different values of the hyperparameters. In Section 5.2.2 we showed how recall, precision and f1-score depend on the number of extracted keywords and concluded 22 keywords is an optimal value for the CRAFT dataset. In this section, we investigate the topic size hyperparameter (minimal number of papers in one cluster) for BERTopic and the number of topics hyperparameter for LDA. In Section 5.2.2, the topic size for BERTopic was fixed to 3, since CRAFT is a small dataset, so BERTopic clusters cannot be large. In the case of LDA, we used a number of topics set to 9, since it was the number detected by BERTopic (so LDA should have the same number of topics to make the comparison reliable).

The dependence between topic size and metrics for BERTopic is visible in Figure 13. In turn, the dependence between a number of topics and metrics for LDA is depicted in Figure 14. For both BERTopic and LDA, the number of extracted keywords was set to 22.

One can conclude that there is no significant difference in metrics values for different hyperparameters neither for BERTopic nor for LDA.

Ground truth mentions	BERTopic keywords
sensory hair cell, supportive cell, cell differentiation, gene expression, sensory system development, sensory organ development, sensory perception of sound, biological regulation, Notch signaling pathway, lateral inhibition, cell cycle, Mammalia, protein jagged-1, delta-like protein 1, protein jagged-2, transcription factor SOX-2, cyclin-dependent kinase inhibitor 1B, gene, embryo, internal ear, sense organ, macula, ear, cochlea, utricle of membranous labyrinth, macula of saccule of membranous labyrinth, spiral organ of cochlea, crista ampullaris	mouse, cell, gene, mutant, expression, Abstract, development, individual, sensory, mutation, transcription, factor, show, phenotype, protein, hair, model, antioxidant, ear, region, allele, 1

Figure 11: LDA keywords extraction (good result).

Ground truth mentions	BERTopic keywords
biological pigment, aging, circadian rhythm, biological regulation, Mus, Mammalia, species, Homo sapiens, gene, allele, eye	mouse, cell, receptor, strain, differentiation, IOP, signaling, D2, bone, gene, function, chromosome, Abstract, apoptotic, time, SirT1, dopamine, development, Ptdsr, response, taste, phenotype

Figure 12: LDA keywords extraction (bad result).

There is also no clear dependence visible between the metrics and hyperparameters values. The best topic size for BERTopic is 6, and the best number of topics for LDA is 7. These results suggest that the influence of hyperparameters is not a decisive factor. However, it is worth using the best

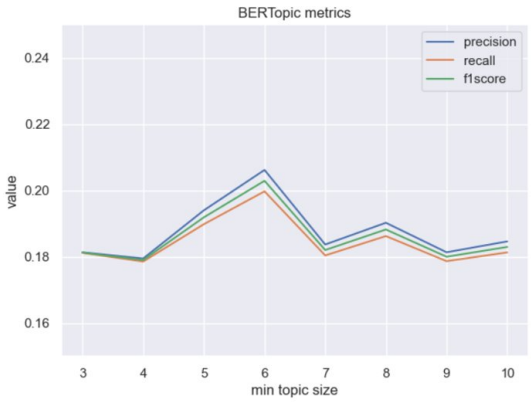

Figure 13: Topic size influence for BERTopic metrics.

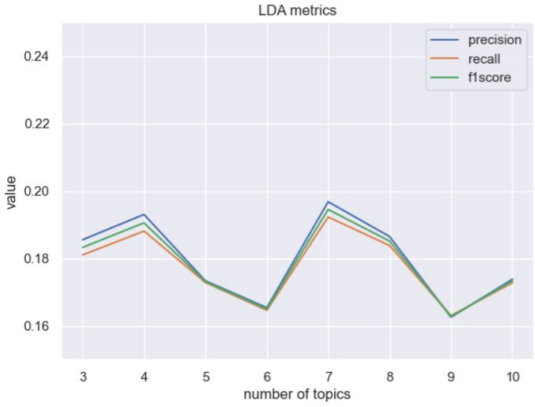

Figure 14: Number of topic influence for LDA metrics.

hyperparameter values in the final pipeline, so we trained a pipeline for the values obtained in the analysis and compared the results.

5.3 Ontologies tagging

5.3.1 NCBO tagger

The NCBO tagging process was similar to the one performed in Project 1. We again queried REST API for tags from specific ontologies. We needed to modify the query since MedMentions ontologies were different than CRAFT ones. In CRAFT, we queried for tags from CHEBI, CL, GO, MONDO, MOP, NCBITaxon, PR, SO, and UBERON.

5.3.2 Embedding tagger

We have used pre-trained bioBERT from the transformers library (Wolf et al., 2020). We calculated the embeddings of keywords extracted by BERTopic and LDA and concept names from nine ontologies (CHEBI, CL, GO, MONDO, MOP,

Keyword	Tagged concept name
sam68	bacterium mgm06b
taste	throat
PAX6	protein PAXX
differentiation	regeneration
mid1	midbody
marrow	bone marrow
D2	DG
a7	azine
mouse	cell
channel	channel protein
neural	processed
genetic	engineered
recombination	recombinational repair
hair	skull
individual	single
pulmonary	chest
olfactory	olfactory pit
photoreceptor	photoreceptor cell
annexin	annexin A
embryonic	embryonic head

Table 3: Table with chosen keywords paired with concepts.

NCBITaxon, PR, SO, UBERON). We paired them against each other and assigned a concept to every keyword, choosing a concept with maximum cosine similarity. Examples of matched pairs are shown in Table 3. These are the top 20 examples (according to cosine similarity) whose similarities are not 1 (keyword is equal to concept name). As we can see, those tagged pairs are rather low quality. We felt the urge to improve our methods so we also tried to pair keywords and concepts against each other choosing concepts with the minimum euclidean distance between them and keywords. It turned out that using euclidean distance we pair keywords with the different concepts only in 7% of the whole cases.

5.3.3 MedCAT

We attempted to use MedCAT (Kraljevic et al., 2021) to tag articles from the CRAFT dataset to compare with our solutions. During a try to use the MedCAT tutorial to annotate CRAFT articles, it turned out that MedCAT annotates using concepts with UMLS and unfortunately 7 out of 9 ontologies used in CRAFT are not present in UMLS. We could use MedCAT just to tag articles with those two remaining ontologies but we think later comparison with our solutions would not be reliable or trustworthy. Because of these reasons, we've abandoned the idea of using MedCAT. In Figure 23 in the Appendix we can see the results of searching ontologies used in CRAFT in UMLS ontologies. Either no results are returned or the results do not match our query.

5.4 Disambiguation

In the Project 1, we performed only simple disambiguation. The initial selection was random, all coexisting keywords had the same impact and we did not take into account the text of the keyword, we are currently examining. Since the disambiguation is an important part of our project (and the one implemented by us from scratch) we decided to further investigate it and introduce some modifications.

5.4.1 Initial sorting

The first modification we introduced is an initial sorting. So far the first "best order of concepts" was created randomly, hence we decided to somehow sort it. Since, as input for disambiguation, the pairs of keywords and concepts are given, and some keywords have multiple concepts assigned, it seems reasonable to initially sort them based on the frequency of each concept occurrence. Therefore, the concepts that appear more often are at the beginning of the initial dictionary (result of the disambiguation - the best keyword-concept assignment). To achieve this, occurrences are counted and the results are sorted while preparing the dictionary for the first iteration. The idea is shown in Figure 15. The mentioned dictionary is an output of the disambiguation process. For each keyword (treated as a key in the dictionary), a dictionary of concepts and their distances (from other keywords, calculated in each iteration) is given. The concepts for the keyword CELL (for which we are performing disambiguation) are given. The concept *cell* is the most frequent one. Therefore in the result dictionary, it is placed in the first position (order from top to bottom shown in Figure 16).

```
07: ['CELL', 'cell']
08: ['CELL', 'cell']
09: ['CELL', 'Fully Formed Anatomi... Structure']
10: ['CELL', 'cellular anatomical structure']
11: ['CELL', 'independent continuant']
12: ['CELL', 'anatomical entity']
13: ['CELL', 'Anatomical Structure']
14: ['CELL', 'Cell']
15: ['CELL', 'cell']
```

Figure 15: The concepts for the keyword CELL

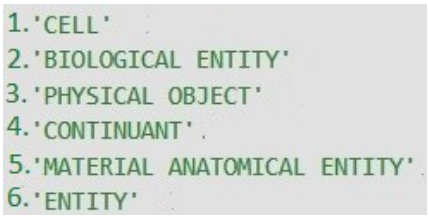

Figure 16: The result dictionary for the keyword CELL

5.4.2 Weighted voting

The next novelty, that we want to introduce is weighted voting. Up to now, each keyword had the same impact on the final result (disambiguation for a given keyword is performed with the context of other keywords - their tags, more in the Appendix). We think it is rational to let the more critical keywords have a bigger impact. The importance of the keywords was taken from the results obtained by *LDA* A.1.1 and *BERTopic* A.1.2. In each iteration, while deciding on the best keyword the votes from the other keywords are weighted based on their importance. In Figure 17 the example weights of keywords are shown.

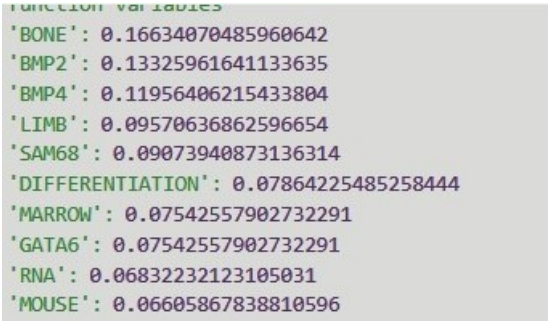

Figure 17: The weights of each keyword

5.4.3 Forced order

After examining the aforementioned options, we detected a phenomenon. Even though, one of the concepts is identical to the keyword (example in Figure 15) it is not selected as the most suitable one. We decided to investigate it. The problem lies in the disambiguation process, which is taking into account only concepts of other keywords. To better understand the problem an example can be given. Imagine there are three keywords: DOG, CELL and TIGER. If each of them has the *"biological entity"* in their concept, this concept would be selected as the best-suited one for each of the keywords. This is consistent with the disambiguation assumption (e.t. the concepts of each keyword should be similar) but not with our intuition. In our opinion (supported with experiments) the concept *"cell"* for a keyword CELL is a better one. Keywords DOG and TIGER can also have a common, better-suited concept (f.e. *"mammal"* or *"animal"*), which cannot be chosen for the CELL and hence with a great probability will not be chosen for those two words as well. If we force the algorithm to select the concept *"cell"* for the keyword CELL, there is a bigger probability that the concept "mammal" or "animal" would be chosen for the remaining keywords. Therefore, in this option, the algorithm is forced to choose the concept identical to the keyword as the best one. An example of this option performance is shown in Figure 18 and 19. In Figure 18 most of the keywords have the concept *"object"* assigned. It is a very wide, not descriptive concept. In Figure 19 the assigned concepts are more specific. Some of them are identical with the keywords (forced by the algorithm), but also others are more descriptive (*"material entity"*).

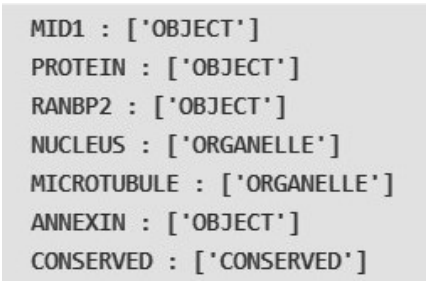

Figure 18: The best keywords without sorting

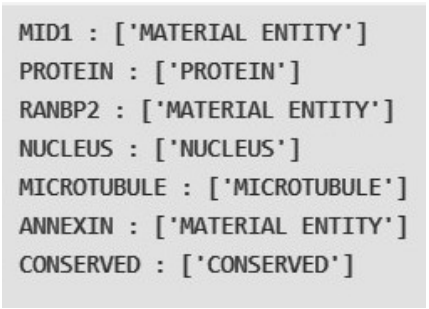

Figure 19: The best keywords with sorting

5.4.4 Disambiguation - summary

To use the new options we performed four types of disambiguation

1. No sorting and no weighting

2. Initial sorting without weighting

3. Initial sorting and weighting

4. Initial sorting, weighting and forcing.

This enables us to compare them and see if our assumptions about sorting and weighing were correct. The examples of disambiguation behaviour, depending on its type are presented in Figures 20 and 21.

```
-----------SIMPLE DISAMBIGUATION-----------------
GENETIC : ['DISEASE CHARACTERISTIC']
MOUSE : ['MATERIAL ENTITY']
MUSCLE : ['MATERIAL ENTITY']
GENE : ['REGION']
DISEASE : ['DISEASE OR DISORDER']

------------INITIAL SORTING-----------------------
GENETIC : ['DISEASE CHARACTERISTIC']
MOUSE : ['MATERIAL ENTITY']
MUSCLE : ['MATERIAL ENTITY']
GENE : ['REGION']
DISEASE : ['DISEASE OR DISORDER']

-----------INITIAL SORTING + WEIGHTING-------------
GENETIC : ['DISEASE CHARACTERISTIC']
MOUSE : ['MATERIAL ENTITY']
MUSCLE : ['MATERIAL ENTITY']
GENE : ['REGION']
DISEASE : ['DISEASE OR DISORDER']

---------INITIAL SORTING + WEIGHTING + FORCING-----
GENETIC : ['DISEASE CHARACTERISTIC']
MOUSE : ['MATERIAL ENTITY']
MUSCLE : ['MATERIAL ENTITY']
GENE : ['GENE']
DISEASE : ['DISEASE OR DISORDER']
```

Figure 20: Example without big difference

In Figure 20 it can be seen, that neither sorting nor weighting made difference in the disambiguation results. The only change was done after enabling forcing option, to the keyword GENE the concept *"gene"* was assigned. On the other hand in Figure 21 the changes in the best-suited concepts assignment can be seen. It's worth highlighting the difference between disambiguation with sorting and weighting but without forcing, and with forcing. In the former version, both keywords HAIR and EAR have the keyword *"continuant"* assigned. While after forcing disambiguation to select concept *"ear"* for the keyword EAR, the concept of HAIR is changed to the *"material entity"*. This shows, that forcing influences also the choices of concepts for other keywords. In Figure 22 the weights of the keywords can be seen, the keyword EAR has the biggest influence and hence this change was so influential.

```
-----------SIMPLE DISAMBIGUATION-----------------
HAIR : ['MATERIAL ENTITY']
CELL : ['BIOLOGICAL ENTITY']
TBX15 : ['MATERIAL ENTITY']
EAR : ['MATERIAL ENTITY']
PENDRIN : ['MATERIAL ENTITY']
COCHLEA : ['MATERIAL ENTITY']

------------INITIAL SORTING-----------------------
HAIR : ['MATERIAL ENTITY']
CELL : ['BIOLOGICAL ENTITY']
TBX15 : ['MATERIAL ENTITY']
EAR : ['MATERIAL ENTITY']
PENDRIN : ['MATERIAL ENTITY']
COCHLEA : ['MATERIAL ENTITY']

-----------INITIAL SORTING + WEIGHTING-------------
HAIR : ['CONTINUANT']
CELL : ['CONTINUANT']
TBX15 : ['MATERIAL ENTITY']
EAR : ['CONTINUANT']
PENDRIN : ['MATERIAL ENTITY']
COCHLEA : ['CONTINUANT']

---------INITIAL SORTING + WEIGHTING + FORCING-----
HAIR : ['MATERIAL ENTITY']
CELL : ['CELL']
TBX15 : ['MATERIAL ENTITY']
EAR : ['EAR']
PENDRIN : ['PENDRIN']
COCHLEA : ['COCHLEA']
```

Figure 21: Example with changes

```
{'EAR': 0.08950458480353375,
 'SENSORY': 0.08130253977745765,
 'CELL': 0.06981591322762211,
 'HAIR': 0.06967881078425237,
 'TBX15': 0.05420145019839328,
 'EXPRESSION': 0.051496847594213704,
 'COCHLEA': 0.04863097864826702,
 'DORSOVENTRAL': 0.04863097864826702,
 'MUTANT': 0.04054045491499,
 'PENDRIN': 0.03690696158244563}
```

Figure 22: The weights used in the example from Figure 21

To sum up, we see the profits of using the disambiguation technique. In many cases, it led to the final metrics increase. The presented modifications also helped with achieving better results. Concerning the detected downsides, the main one was described in subsection 5.4.3. Another one was connected with the calculated embeddings. Since they can only be calculated for a single word, we averaged over longer concepts. As a result, some concepts are claimed to be more similar, than they really are (f.e. *biological entity* and *material entity*).

5.5 Final results

To compare tagging methods (in the entire algorithm pipeline), we used two metrics: precision, recall, and F1 score, which are defined as follows:

$$precision = \frac{TP}{TP + FP} \tag{4}$$

$$recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{6}$$

Where:

- True positives (TP) - number of cases when our annotation matches annotation from the golden standard dataset
- False positives (FP) - number of cases when our annotation does not match annotation from the golden standard dataset
- False negatives (FN) - number of cases when annotation from the golden standard dataset is not present in our annotations

It is hard to come up with a way to calculate the number of true negatives. That is why we choose metrics that do not rely on this particular number.

In Table 4, we provided the evaluation of our methods - three metrics: recall, precision, and F1 score. The results obtained for NCBO tagger (Jonquet et al., 2009) were not satisfactory for either of the chosen hyperparameters (the possible reasons are the naivety in NCBO implementation and the difficulties in a disambiguation process - described in A.3). The better results are obtained by embedding tagger (A.2.4). The best one (taking the F1 score as a measure) is a result of the BERTopic (Grootendorst, 2022b) tagging with 22 keywords and a minimum topic size of 6. These values reflect the best values obtained during hyperparameters tuning. The similarity calculations, for this example, were done based on the Euclidean distance.

6 Conclusions and future works

In this project we developed techniques do evaluate Project 1. Apart from evaluating the entire pipeline (keywords extraction, ontologies tagging and disambiguation) at once, we also evaluated keywords extraction step separately, as this part is difficult to evaluate based only on ground-truth annotations. In this case, we performed both quantitative study, focusing on precision, recall, f1-score and qualitative research aiming to explain how BERTopic and LDA find the most important words. We investigated the influence of hyperparameters. Finally, we tested and compared different variants of disambiguation techniques.

Keywords extraction evaluation showed that models used for extracting keywords work with good performance, and extracted keywords are relevant. The examples we included in this report showed that based on even a small number of keywords, the subject of a particular paper could be deduced. We again confirmed that the evaluation of the SLR process is a very challenging task. We indicated the examples in which the number of true positives for BERTopic was zero, but, despite it, extracted keywords were correct, and it was easy to find out what is the paper topic. The reason for it is that keywords extraction is a task that has multiple correct solutions, and even ground truth created by humans represents just one of them. Moreover, they are entirely subjective - it is illustrated by the examples we showed (choice of mentions in Figure 7 and the *mouse* keyword, for instance).

The disambiguation algorithm was implemented by us from scratch. It turned out that the proposed solution is not sufficient for this challenge. Even though, we proposed a few different disambiguation options, the received results were not satisfactory. We believe, that the algorithm was too simple and it made us of too many simplifications. One of the biggest problem, connected to disambiguation, was the lack of ability to get embeddings from the whole phrases (we could only retrieve the single word embedding), consequently longer phrases needed to be averaged and consequently lost part of the meaning.

In the case of the tools we used, the future works can include trying different taggers (or trainning exisitng apporaches, like MedCAT, on the ontologies from CRAFT datasets) and more advanced disambiguation techniques. Nevertheless, the most challanging part of our project was the lack of suitable dataset (either inconsistent MedMentions dataset or small CRAFT). That is why we claim that the most important task to do in the domain of automating SLR is to create a sufficiently large and coherent dataset, that would

Keyword extraction	Number of keywords	Min topic size/ number of topics	Tagger	Similarity type	Precision	Recall	F1
BERTopic	10	3	Embedding	cosine	4.12	8.21	5.49
BERTopic	10	3	Embedding	distance	4.44	8.49	5.83
BERTopic	22	6	Embedding	cosine	7.35	6.86	7.1
BERTopic	22	6	Embedding	distance	7.35	6.79	**7.06**
LDA	10	9	Embedding	cosine	4.92	9.71	6.53
LDA	10	9	Embedding	distance	5.18	9.9	6.8
LDA	22	7	Embedding	cosine	5.55	5.38	5.47
LDA	22	7	Embedding	distance	5.82	5.52	5.67
BERTopic	22	6	NCBO	-	0.07	0.1	0.08
BERTopic	22	7	NCBO	-	0.05	0.08	0.06
BERTopic	10	3	NCBO	-	0.05	0.16	0.08
BERTopic	10	9	NCBO	-	0.05	0.12	0.07

Table 4: Table showing results of different keyword extraction and tagging methods in CRAFT (Cohen et al., 2017).

allow the baseline comparisons of different algorithms. It is important to tag such a dataset in a honest way, include all relevant keywords or phrases and take care of defining which ontologies were used to tag it. The key aspect is to ensure that all papers are tagged according to the same tagging rules and scheams. With such a dataset created, the evaluation process of SLR will be both easier and more relevant.

References

Dimitra Alexopoulou, Bill Andreopoulos, Heiko Dietze, Andreas Doms, Fabien Gandon, Jörg Hakenberg, Khaled Khelif, Michael Schroeder, and Thomas Wächter. 2009. Biomedical word sense disambiguation with ontologies and metadata: automation meets accuracy. *BMC Bioinformatics*, 10(1):28, Jan.

Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. 2000. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, May.

Silvia Bindelli, Claudio Criscione, Carlo Curino, Mauro Drago, Davide Eynard, and Giorgio Orsi. 2008a. Improving search and navigation by combining ontologies and social tags. pages 76–85, 11.

Silvia Bindelli, Claudio Criscione, Carlo Curino, Mauro Drago, Davide Eynard, and Giorgio Orsi. 2008b. Improving search and navigation by combining ontologies and social tags. pages 76–85, 11.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022, mar.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning, URL:* `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html`.

K Bretonnel Cohen, Arrick Lanfranchi, Miji Joo-young Choi, Michael Bada, William A Baumgartner, Natalya Panteleyeva, Karin Verspoor, Martha Palmer, and Lawrence E Hunter. 2017. Coreference annotation and resolution in the colorado richly annotated full text (craft) corpus of biomedical journal articles. *BMC bioinformatics*, 18(1):1–14.

Alexander D. Diehl, Terrence F. Meehan, Yvonne M. Bradford, Matthew H. Brush, Wasila M. Dahdul, David S. Dougall, Yongqun He, David Osumi-Sutherland, Alan Ruttenberg, Sirarat Sarntivijai, Ceri E. Van Slyke, Nicole A. Vasilevsky, Melissa A. Haendel, Judith A. Blake, and Christopher J. Mungall. 2016. The cell ontology 2016: enhanced content, modularization, and ontology interoperability. *Journal of Biomedical Semantics*, 7(1), July.

Karen Eilbeck, Suzanna E Lewis, Christopher J Mungall, Mark Yandell, Lincoln Stein, Richard

Durbin, and Michael Ashburner. 2005. *Genome Biology*, 6(5):R44.

Maarten Grootendorst. 2022a. Bertopic - pypi, url: https://pypi.org/project/bertopic/.

Maarten Grootendorst. 2022b. Bertopic: Neural topic modeling with a class-based tf-idf procedure.

Melissa A Haendel, James P Balhoff, Frederic B Bastian, David C Blackburn, Judith A Blake, Yvonne Bradford, Aurelie Comte, Wasila M Dahdul, Thomas A Dececchi, Robert E Druzinsky, Terry F Hayamizu, Nizar Ibrahim, Suzanna E Lewis, Paula M Mabee, Anne Niknejad, Marc Robinson-Rechavi, Paul C Sereno, and Christopher J Mungall. 2014. Unification of multi-species vertebrate anatomy ontologies for comparative biology in uberon. *Journal of Biomedical Semantics*, 5(1):21.

Janna Hastings, Gareth Owen, Adriano Dekker, Marcus Ennis, Namrata Kale, Venkatesh Muthukrishnan, Steve Turner, Neil Swainston, Pedro Mendes, and Christoph Steinbeck. 2015. ChEBI in 2016: Improved services and an expanding collection of metabolites. *Nucleic Acids Research*, 44(D1):D1214–D1219, October.

Clement Jonquet, Nigam Shah, Cherie Youn, Mark Musen, Chris Callendar, and Margaret-Anne Storey. 2009. Ncbo annotator: Semantic annotation of biomedical data. *ISWC*, 01.

Clement Jonquet, Nigam Shah, Cherie Youn, Mark Musen, Chris Callendar, and Margaret-Anne Storey. 2021. Ncbo annotator rest api, url: https://bioportal.bioontology.org/annotator, 01.

İlknur Karadeniz and Arzucan Özgür. 2019. Linking entities through an ontology using word embeddings and syntactic re-ranking. *BMC Bioinformatics*, 20(1):156, Mar.

Zeljko Kraljevic, Thomas Searle, Anthony Shek, Lukasz Roguski, Kawsar Noor, Daniel Bean, Aurelie Mascio, Leilei Zhu, Amos A Folarin, Angus Roberts, Rebecca Bendayan, Mark P Richardson, Robert Stewart, Anoop D Shah, Wai Keong Wong, Zina Ibrahim, James T Teo, and Richard J B Dobson. 2021. Multi-domain clinical natural language processing with MedCAT: The medical concept annotation toolkit. *Artif. Intell. Med.*, 117:102083, July.

Robert Leaman and Zhiyong Lu. 2016. TaggerOne: joint named entity recognition and normalization with semi-Markov Models. *Bioinformatics*, 32(18):2839–2846, 06.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *CoRR*, abs/1901.08746.

Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11), mar.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction.

Sunil Mohan and Donghui Li. 2019. Medmentions: A large biomedical corpus annotated with umls concepts.

Molecular process ontology (mop). https://github.com/rsc-ontologies/rxno. Accessed: 2023-01-03.

Darren A. Natale, Cecilia N. Arighi, Judith A. Blake, Jonathan Bona, Chuming Chen, Sheng-Chih Chen, Karen R. Christie, Julie Cowart, Peter D'Eustachio, Alexander D. Diehl, Harold J. Drabkin, William D. Duncan, Hongzhan Huang, Jia Ren, Karen Ross, Alan Ruttenberg, Veronica Shamovsky, Barry Smith, Qinghua Wang, Jian Zhang, Abdelrahman El-Sayed, and Cathy H. Wu. 2016. Protein ontology (PRO): enhancing and scaling up the representation of protein entities. *Nucleic Acids Research*, 45(D1):D339–D346, November.

Claire Nédellec, Robert Bossy, Estelle Chaix, and Louise Deléger. 2018. *Text-mining and ontologies: new approaches to knowledge discovery of microbial diversity*. May.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy, August. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Conrad L Schoch, Stacy Ciufo, Mikhail Domrachev, Carol L Hotton, Sivakumar Kannan, Rogneda Khovanskaya, Detlef Leipe, Richard Mcveigh, Kathleen O'Neill, Barbara Robbertse, Shobha Sharma, Vladimir Soussov, John P Sullivan, Lu Sun, Seán Turner, and Ilene Karsch-Mizrachi. 2020. NCBI taxonomy: a comprehensive update on curation, resources and tools. *Database*, 2020, January.

Raymon van Dinter, Bedir Tekinerdogan, and Cagatay Catal. 2021. Automation of systematic literature reviews: A systematic literature review. *Information and Software Technology*, 136:106589.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October. Association for Computational Linguistics.

Appendices

A Methods, techniques, devices to be used in research

In this section, we describe methods and techniques used in developing the project solution.

A.1 Keyword extraction

To detect the main concepts of the given documents, we decided to use Topic Modeling. It is an unsupervised machine learning method, that scans a set of documents and clusters them into groups represented by similar abstract topics. The conventional technique LDA (Blei et al., 2003) treats a document as a bag-of-words. Consequently, it loses the context and the order of the words. To prevent order loss and profit from the context of the given word, text embedding techniques have been used in various tasks. In recent years they became popular in the topic modeling field. Therefore, we decided to use BERTopic (Grootendorst, 2022b). However, we compare its performance with LDA.

A.1.1 LDA

The Latent Dirichlet Allocation is a generative probabilistic model for finding hidden topics in the given corpora, proposed in (Blei et al., 2003). It makes a few assumptions:

1. topics are the statistically significant words in given corpora,
2. documents are a mixture of topics,
3. topics are a mixture of words.

Based on them, LDA calculates the probability density of topics in the document.

Before performing the algorithm pre-processing is needed, words need to be tokenized and a number of expected topics need to be given (in this work it is going to be denoted as Q). After that, the word-document matrix is created. This matrix is then divided into two matrices: document-topic and topic-word one.

LDA is an iterative process. In the first iteration, the randomly selected topics are assigned to each word. After that, LDA tries to optimize the results. In order to do this, it examines each word separately. Assuming that all assigned topics, apart from the current one, are correct. LDA

tries to find the best topic for a given word. To do this it calculates 2 probabilities;

1. p1: proportion of words in a given document with a given topic (q),
2. p2: proportion of the documents in which the word (w) has the topic q assigned.

Using those probabilities, it detects the most relevant topic for a given the word and reassigns it.

For each word in each document, the procedure is repeated until a steady solution is found. In the end, the list of Q tuples containing the topic number and the list of most informative terms with their probability is given. LDA does not interpret topics, this step needs to be performed manually (it only provides the topic number and the informative words, the user needs to add the topic description/ name if needed).

A.1.2 BERTopic

In this project, we use the (Grootendorst, 2022a) BERTopic framework implementation.

To generate topic representation, BERTopic goes through 3 main steps.

First, it embeds documents in order to create their representation in vector space and compare their semantic meaning. As a default, it uses Sentence-BERT (SBERT) framework (Reimers and Gurevych, 2019a), which enables converting sentences into vector representation using a pre-trained language model. SBERT is an extension of the traditional BERT model, for which calculating the sentence probability is a very time-consuming task. As authors of the (Reimers and Gurevych, 2019b) claim, by adding the pooling operation at the output of the BERT, the time of finding the most similar sentence pair in a collection of 10000 sentences was reduced from 65 hours to 5 seconds. Therefore, the BERTopic framework, by default, makes use of the SBERT model. It also allows using other pre-trained sentence embedding or custom models.

Subsequently, it performs clustering. Due to high space dimensionality, calculating the distance might become ill-defined. Therefore, to reduce dimensionality UMAP (McInnes et al., 2018) algorithm is used. The reduced embeddings are clustered using HDBSCAN (McInnes et al., 2017). The BERTopic framework allows changing both dimensionality reduction and clustering algorithms. The aforementioned techniques are used as default methods.

The last step is finding the topic representation. As a default, the modified TF-IDF procedure is used. The original procedure combines term, and inverse document frequency:

$$W_{t,d} = tf_{f,d}\dot{l}og(\frac{N}{df_t}), \tag{7}$$

where $tf_{t,d}$ is the frequency of the term t in document d, N is the number of documents and df_t is a document frequency that shows how much information the term provided in the document. In BERTopic this procedure is generalized to clusters of documents. Firstly, all documents in the cluster are concatenated, and then TF-IDF is modified and obtained by the formula:

$$W_{t,c} = tf_{f,c}\dot{l}og(1 + \frac{A}{tf_t}), \tag{8}$$

where tf_t, c is a frequency of the term t in the class c. C is concatenated into one document collection of the documents from the same cluster. Tf_t is a class frequency, measuring how much information the term provides to a class. By using the modified TF-IDF formula the importance of the words in a cluster, rather then in the document, is modeled.

A.2 Tagging tools

Keywords, extracted in the previous step, might incorporate different names to describe the same concepts. Therefore, to make use of them it is essential to perform mapping into existing ontologies. The ontology provides the standardized, homogenous, and informative concept, that describes the given keyword. The task of tagging the word with an entity existing in the ontology is called entity normalization, entity grounding, or entity categorization. As an example of the biomedical entity grounding, we will use the Onto-Biotope Ontology (Nédellec et al., 2018). Given the words "pediatric", "respiratory" and "children less than 2 years", we aim to the appropriate tags in the ontology. For the first two words, the task is relatively simple. "Pediatric" ought to be linked to "pediatric patient" and "respiratory" to the "respiratory tract part". Both of those examples are lexically similar. In the last case, the linking is not that trivial. "Children less than 2 years" should be tagged as a "pediatric patient", even though the lexical similarity does not exist. The given example was derived from (Karadeniz and Özgür, 2019). Moreover, in the biomedical domain, the number of se-

mantic categories is greater than the entities mentioned in available training data sets. For example, Onto-Biotope ontology consists of 2221 categories, while only 747 of them were mentioned in the training data set. Therefore, we decided to use unsupervised annotation techniques. In this section, we described two approaches tested in our solution. The first is NCBO tagger, which annotates data based mostly on direct string matching. The second, which uses word embeddings to link entities using word.

A.2.1 MedCAT

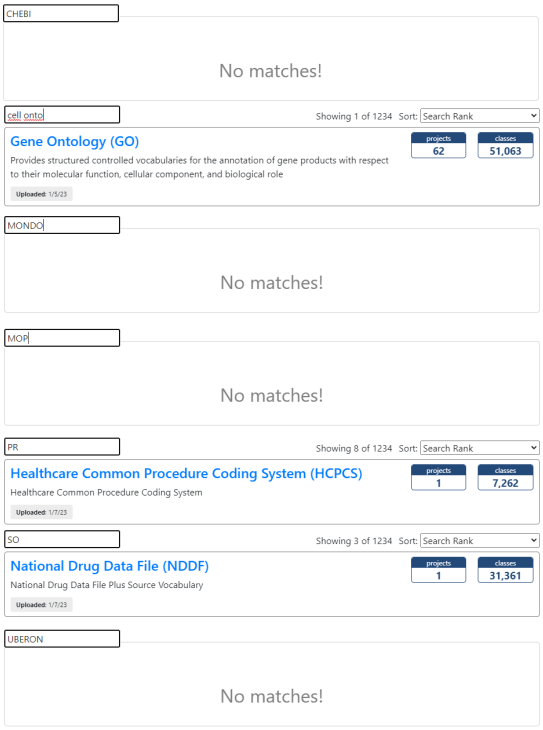

Figure 23: Results of searching ontologies used in CRAFT in UMLS ontologies.

A.2.2 NCBO tagger

NCBO tagger (Jonquet et al., 2009) is a result of an initiative to construct a solution for annotating biomedical data with the use of a great number of ontologies. At the time of releasing the solution it used over 200 ontologies and this number is constantly increasing.

The way NCBO tagger works is simple, yet in many cases powerful enough. It uses a few steps to tag each token of an input free-text.

First of all, a direct string matching is performed. The dictionary of ontologies concepts is used for this purpose. It is constructed by pooling all concept names or other string forms (synonyms, labels) that syntactically identify concepts. Then, tokens from the input string are matched to this dictionary entries.

Second step is performed by *is_a transitive closure*, which aims to explore the relations in ontologies, namely for a given matched concept it searches its subsequent ancestors in the parent-child hierarchy and can match them as tags for a given token as well. The number of ancestors to look through is paramterizable.

Next, an *ontology-mapping component* tries to find relations between different ontologies, e.g., when a given concept is matched for a token, it can be linked to a respective concept in another ontology, and the ontology can also be traversed.

As a result, NCBO produces quite a lot of tags for each input text token. Our first trials showed that they are usually relevant, however, often to many of them is generated. Hence the need to select the most suitable tag and possibly perform disambiguation.

A.2.3 BIOBert

BERT (Vaswani et al., 2017) (Bidirectional Encoder Representations from Transformers) is a language processing model developed by Google that has been widely used for natural language processing (NLP) tasks such as text classification, language translation, and named entity recognition. It is based on the transformer architecture, which uses self-attention mechanisms to process input sequences in parallel, allowing the model to effectively handle long-range dependencies in language and achieve strong performance on a variety of NLP tasks.

BioBERT (**?**) is a variant of the BERT model that has been specifically designed for natural language processing (NLP) tasks in the biomedical domain. It was trained on a large dataset of biomedical literature and has been shown to perform well on a variety of NLP tasks in the biomedical domain.

A.2.4 Words embedding tagger

The approach mentioned in (Karadeniz and Özgür, 2019) is based on the assumption that semantically similar words have similar vectors in the embedded space.

Before computing the word embedding vectors, the preprocessing is performed. The words need

to be free from stop words, and non-ASCII characters.

Next, the word vectors are calculated, using the pre-trained model. For multi-word entities, each word is transformed separately and the average vector is calculated. After the conversion of both tagged data and ontology concepts, their similarity is measured. We use cosine similarity which is calculated by the given equation:

$$cosine\_similarity = \frac{AB}{\parallel A \parallel \parallel B \parallel}, \quad (9)$$

where A and B are the vectors. After performing those steps, the list of closest ontologies concepts is given.

A.3 Words tags disambiguation

In free-text data, the same word can occur in different contexts and with different meanings. For example, if working with data containing information about wines, *Burgundy* can refer to the name of the wine or the region in France (Bindelli et al., 2008b). Hence, it should be decided whether to tag *Burgundy* with *Wine name* or *Country Region*. This information should be based on the context of a tagging word in an input text. If it comes to medical data, the term *blood pressure* can have three senses, namely *organism function*, *diagnostic procedure* and *laboratory or test result* (Alexopoulou et al., 2009).

We propose a method inspired by Alexopoulou et al. (2009). It is based on selecting the sense of a given word (or in general token) that is the closest to senses of other words appearing in the context. In the following subsections we describe the original Closest Sense method (sentence-based) and the modification than was incorporated in our solution.

A.3.1 Sentence-based Closest Sense method

Let us suppose that we want to tag tokens in the sentence: *I also tracked lipid profiles, HBA1C, blood pressure, body mass index, hostility and nicotine use*. As mentioned above, *blood pressure* can have multiple senses since it is ambiguous - three tags are possible (assuming they are concepts of some ontology): *organism function*, *diagnostic procedure* and *laboratory or test result*.

To decide which tag should be assigned to *blood pressure*, we explore tags of other words appearing in the sentence. Let us assume that the senses of the occurring terms are *laboratory procedure* (lipid profile), *gene or genome* (HBA1C), *diagnostic procedure* (body mass index), *mental process* (hostility) and *organic chemical* (nicotine). Then for blood pressure, we choose the sense that is on average closer to the senses of the co-occurring terms than the other candidate senses.

What the *closeness* means can be treated in various ways. For example, semantic distances utilizing the ontologies (like subsumption distance or subtype-aware signature distance) can be used (Alexopoulou et al., 2009). The other way could be to incorporate words embedding and investigate cosine similarity.

A.3.2 Keywords-based Closest Sense method

Since our task aims to tag keywords instead of particular words in free text, we plan to modify the Closest Sense algorithm.

First of all, for a given ambiguous keyword, we are going to treat other keywords extracted for a given text as the context, instead of words that occur in the same sentence.

Secondly, in the case of our problem, each keyword is ambiguous on a similar level (all keywords will have numerous candidate tags assigned). This is a difference in regards to what Alexopoulou et al. (2009) explored: they assumed only a given word in a sentence is ambiguous while other words have correctly assigned tags. That is why we propose the following iterative procedure: given a set K of keywords k_j, $j = 1, \ldots, |K|$ for a given document and sets $|T_j|$ of candidate tags: t_{j,i_j} for j-th keyword, $i_j = 1, \ldots, |T_j|$ perform *max_iter* times:

1. Take subsequent keyword k_j and assume this keyword is ambiguous, while all other keywords have correct tags assigned (take first tag in the candidate list for these keywords).

2. For each candidate tag t_{j,i_j} calculate the similarity distances to other keywords tags and sort the list of tags from $|T_j|$ by decreasing similarity distance. As a result, at the top of the list we have the best tag for k_j according to the current state.

3. Go to point 1. taking next keyword.

After max_iter iterations of above points, each k_j will be considered max_iter times. This heuristic can help to choose keywords tags according to the context of the entire document.