Topical Classification of Food Safety Publications with a Knowledge Base

Piotr Sowiński[1,2], Katarzyna Wasielewska-Michniewska[2], Maria Ganzha[1,2], and Marcin Paprzycki[2]

[1] Warsaw University of Technology, Warsaw, Poland
piotr.sowinski@mini.pw.edu.pl, maria.ganzha@pw.edu.pl
[2] Systems Research Institute, Polish Academy of Sciences
Warsaw, Poland
{katarzyna.wasielewska,marcin.paprzycki}@ibspan.waw.pl

Abstract. The vast body of scientific publications presents an increasing challenge of finding those that are relevant to a given research question, and making informed decisions on their basis. This becomes extremely difficult without the use of automated tools. Here, one possible area for improvement is automatic classification of publication abstracts according to their topic. This work introduces a novel, knowledge base-oriented publication classifier. The proposed method focuses on achieving scalability and easy adaptability to other domains. Classification speed and accuracy are shown to be satisfactory, in the very demanding field of food safety. Further development and evaluation of the method is needed, as the proposed approach shows much potential.

Keywords: publication classification, ontology, named entity linking, systematic literature review, food safety

1 Introduction

In modern science we are confronted with a fast-growing body of publications, where finding the "relevant ones" is becoming increasingly challenging. To address the problem, evidence-based decision making (EBDM) [3] was proposed, where one of the forms of research used is the systematic review (SR), which uses scientifically rigorous methods to identify, select, assess, and summarize multiple works [20]. Systematic reviews are increasingly used to summarize the state-of-the-art and to guide the direction of future research [2]. While SRs are very reliable, due to the required time and expertise, producing them quickly proved to be a challenge [26]; e.g. a study is typically included in a systematic review 2.5–6.5 years after publication [14]. Hence, the need for automation. While some parts of an SR require creativity and expertise, others can be automated [26].

This work discusses the use of semantic technologies and natural language processing to aid automation of SR in the field of food safety. More specifically, the described system facilitates relevance screening by classifying articles according to their topic(s). Having an article described by its title, abstract, and,

optionally, unstructured keywords, the aim is to assign to it topics drawn from a knowledge base (KB). Here, potential advantage of using a semantic KB is explored, because of the meaning and the context provided by the ontology [22].

The proposed solution was evaluated in the food safety domain, which presents a unique challenge in automatic SR, as it involves a wide variety of topics from multiple fields of science necessitated by the integration of heterogeneous sources of information [4, 27]. On the other hand, there are numerous ontologies available for biology, chemistry, and medicine domains [28], which can be incorporated to reason about food safety.

2   Related works

Food safety is a relatively unexplored domain in text classification-oriented research. However, documents in this domain draw terminology from numerous biomedical sciences. Thus, one may expect to encounter similar characteristics to those present in related areas. For diseases and drugs, issues with entity linking usually stem from the many ways in which a single entity can be represented in text [18]. There are examples of similar morphology but different meaning: "ADA-SCID (adenosine deaminase deficiency)" and "X-SCID (X-linked combined immunodeficiency diseases)". There are also examples of entirely different morphology, but similar semantic meaning, e.g. "kaplan plauchu fitch syndrome" and "acrocraniofacial dysostosis". Moreover, the numerous obscure abbreviations can vary between publications, bringing importance to the context. Finally, very few labeled datasets are available, making supervised methods unfeasible [30].

On the other hand, in biomedical sciences there are many well-structured ontologies that may be used in entity linking [1, 30]. There is a relatively large body of research concerning entity linking for medical reports and publications [18]. However, these works usually focus on linking text to thesauri or taxonomies, such as MeSH[3] that are less expressive than full ontologies. Moreover, these methods are often evaluated using benchmark datasets that provide researchers with training data for supervised models. Nonetheless, even very sophisticated methods, based on BERT, can struggle to correctly perceive context and resolve ambiguous entities. Especially hard are situations where the model is asked to identify terms that occur rarely within the training corpus [10]. Finally, modern supervised approaches have very high hardware requirements.

Furthermore, there a a few solutions addressing the classification problem in the biomedical domain. Here, we outline most relevant to the presented work.

NCBO Annotator[4] is an ontology-based tool for entity linking in biomedicine. Being a part of the BioPortal[5], it supports all ontologies published there. The Annotator uses basic string matching to find entities with names similar to those occurring in the text. Results are further enhanced with concept expansion and ontology mappings [15]. Here, the method is unsupervised and can handle

---

[3] https://www.nlm.nih.gov/mesh/introduction.html
[4] https://bioportal.bioontology.org/annotator
[5] https://bioportal.bioontology.org/

hundreds of ontologies at the same time. However, the system lacks the ability to handle morphological variants and more complicated disambiguation cases.

Neural Concept Recognizer [1] links entities from medical ontologies. It is based on the idea that both text and ontologies can be embedded into a vector space and one can compute similarity metrics between vectors to find matching terms. Text is vectorized using word embeddings and a convolutional neural network. Concept embeddings are established with respect to the taxonomical structure of the ontology. Training is performed on an artificially made dataset, derived from labels present in the ontology. The approach allows training the classifier based only on the ontology and allows easy adaptation to a different domain. It is also claimed that the system can handle novel synonyms, contrary to rule- or dictionary-based solutions. However, the authors point to the inability of the model to perceive context and a relatively slow performance.

The CSO Classifier [22] is an automatic classification system for computer science publications. Although it does not focus on the biomedical sciences, its general idea seems applicable to other domains as well. It uses the Computer Science Ontology of research topics, with a taxonomical structure [23]. The system consists of three modules. The syntactic module preprocesses the text and searches for n-grams, with high Levenshtein similarity to terms in the ontology. This is to find topics that are explicitly mentioned in the document. The semantic module aims at retrieving topics that are semantically related, but not explicitly mentioned. Part of speech tagging and a simple grammar rule are used for extracting candidate text spans to be analyzed. Next, word embeddings (via word2vec) are used to find semantically similar terms in the ontology. The most relevant topics are selected by ranking them using a metric that considers the frequency with which they occur in the document and how "diverse" were their mentions. Finally, in postprocessing, the terms most frequently occurring in the corpus are discarded and the remaining list of topics is enhanced by including direct ancestors of the terms from the ontology. According to the authors, the CSO Classifier outperforms other topic detection methods in terms of recall, by also in including topics that are only implied in text but not explicitly stated.

3   Dataset preparation

Let us now describe datasets that were used in our work. For constructing a corpus of food safety publications, two popular sources were used: PubMed[6] and the European Food Safety Authority (EFSA) Journal[7]. For both sources, articles relevant to food safety were found and their metadata retrieved.

3.1   PubMed

To retrieve the metadata from PubMed, the Entrez API client, from the Biopython package, was used [5]. Articles in PubMed are described using MeSH, which

---

[6] https://pubmed.ncbi.nlm.nih.gov/
[7] https://www.efsa.europa.eu/en/publications

through its hierarchical structure groups descriptors into categories. Our search was limited to articles that use the "food safety" MeSH descriptor, and any of its children; 77 348 articles were found, of which 61 941 had available abstracts. Table 1 shows how often descriptors from a given MeSH category were used to point to articles in the corpus. The analysis was performed using SPARQL queries on MeSH and article metadata, imported into a Blazegraph quad store[8].

Table 1. Descriptor usage by category for PubMed articles

Category	# descriptors
Anatomy [A]	**26 050**
including: Fluids and Secretions	4 790
Cells	4 691
Plant Structures	3 770
Digestive System	2 519
Organisms [B]	**119 840**
Diseases [C]	**45 388**
including: Infections	15 672
Animal Diseases	5 046
Digestive System Disorders	4 205
Chemicals and Drugs [D]	**209 783**
Analytical, Diagnostic and Therapeutic Techniques (...) [E]	**101 766**
including: Investigative Techniques	81 670
Psychiatry and Psychology [F]	**5 513**
Phenomena and Processes [G]	**128 724**
Disciplines and Occupations [H]	**12 752**
Anthropology, Education, Sociology (...) [I]	**8 712**
Technology, Industry, and Agriculture [J]	**69 369**
including: Food and Beverages	35 432
Technology, Industry, and Agriculture	31 536
Humanities [K]	**1 079**
Information Science [L]	**6 161**
Named Groups [M]	**9 822**
Health Care [N]	**111 808**
Geographicals [Z]	**30 474**

It can be observed that food science is *not* the dominant area. Even in [A] and [C] categories, terms associated with the digestive system are not the most frequent. Among frequent terms, not related to food science, are: organisms, chemicals, investigative techniques, phenomena and processes, health care, and geo-location. Thus, it is obvious that *food safety touches upon a very diverse set of topics and involves complex relations*. Hence, when choosing ontologies for describing this domain, a very wide spectrum of topics has to be included.

---

[8] https://blazegraph.com/

To identify additional keywords that were not provided explicitly with the publications, named entity linking (NEL) of abstracts, against the UMLS metathesaurus, was performed with the scispaCy [21] library. The used model was based on BERT architecture (`en_core_sci_scibert`), and the linker was targeting a ~785k entity subset of UMLS. Finally, entity occurrence frequency in abstracts was summarized. The method identified 48 083 unique entities in 61 941 abstracts. The results contain some obvious misclassifications – for example 28 253 abstracts are to refer to mental concentration (attention concentration), which is a very uncommon topic in food safety. These occurrences most likely refer to the concentration of substances, which is a frequent topic in food safety literature. Barring the misclassifications, many of the frequently identified keywords refer to investigative techniques, risk assessment, and statistics (e.g. *levels*, *concentration*, *detection*, *contamination*, *association*, *significance*, *prevalence*). Most other entities are typically within the scope of MeSH (biology, chemistry, medicine).

3.2   EFSA

EFSA metadata is available via a free API[9], which allows retrieving abstracts and associated keywords. As the primary focus of EFSA is food safety, all available articles were retrieved. Of 10 088 publications, 6 684 had available abstracts.

Provided keywords are not structured, which prohibits analysis that was performed using MeSH descriptors. On the other hand, these keywords are not restricted by a controlled vocabulary and thus a rudimentary frequency analysis may show often used terms that were not identified previously. Among the most common keywords that are not directly within the scope of food science, are: *risk assessment*, *safety*, *health claims*, *MRL* (maximum residue levels), *QPS* (qualified presumption of safety), *exposure*, *quarantine*, *children*, *data collection*, *confirmatory data*, European Union member states, and various EU legislative documents. The latter appeared in contributions dealing with regulation compliance. It should be noted that nearly 67% of keywords appear only once, while 14% appear twice. These rarely-appearing terms are mostly specific chemicals, processes, and organisms.

Entity linking against UMLS was performed for the retrieved abstracts, using the same method as with the PubMed corpus. For 6 684 abstracts, 13 200 unique entities were found. As in PubMed, there were many misclassified terms that are unlikely to appear frequently in the corpus. The used NEL method is based on simple n-gram vectorization of mentions and lacks the necessary context required to successfully disambiguate some terms. Additionally, it failed to identify terminology specific to the EFSA dataset (e.g. *European Commission*, *MRLS*, *MON*), due to UMLS lacking these terms. On the other hand, typical biomedical terms were identified easily. Often found were references to food products, ingredients, organisms, age groups, chemicals, and diseases.

---

[9] https://openapi-portal.efsa.europa.eu/

3.3   Summary

From the preliminary analysis it is obvious, that dictionaries (ontologies) used in food safety should span a wide scope of biomedical topics. Moreover, the preliminary NEL results obtained using scispaCy strongly suggest that disambiguating of named entities in texts may require the system to better perceive the context in which the term appears and how is it related to other terms.

4   Ontologies

While initial results suggest that the use of semantic technologies could be beneficial to the task of publication classification, there is no public food safety ontology available. However, there are ontologies describing related fields, such as food science. One example is FoodOn – a comprehensive food ontology [9], available through NCBO's BioPortal [28]. FoodOn has numerous mappings to other ontologies in the BioPortal[10], which allows them to be used jointly. FoodOn is also a part of OBO Foundry[11], guaranteeing application of several quality-improving guidelines [25], especially important when using multiple ontologies at a time.

As shown previously, food safety publications touch upon many diverse topics within the biomedical domain. The number of available biomedical ontologies is also high – NCBO's BioPortal claims to host 846 ontologies. Without the help of domain experts, the only viable method for ontology selection (and use) would have to be based on data-driven analysis of topics found in publications. Such analysis can be based on data gathered from PubMed, thanks to the presence of structured MeSH descriptors. Those descriptors can be mapped to related entities in other ontologies, using mappings from the BioPortal.

For each of the 11 693 MeSH descriptors, present in the corpus, mappings to all BioPortal ontologies were retrieved. The initial focus was on ontologies from the OBO Foundry. Actively maintained ontologies with comprehensive documentation were prioritized. Additionally, several upper level ontologies were included (e.g., BFO, IAO, RO). The selected OBO ontologies are listed below, ordered alphabetically by their abbreviations.

- Anatomical Entity Ontology (**AEO**)[12]
- Agronomy Ontology (**AGRO**)[13]
- Apollo Structured Vocabulary (**APOLLO-SV**)[14]
- Basic Formal Ontology (**BFO**)[15]
- BRENDA tissue / enzyme source (**BTO**)[16]

---

[10] https://bioportal.bioontology.org/ontologies/FOODON/?p=mappings
[11] http://obofoundry.org/
[12] http://www.obofoundry.org/ontology/aeo.html
[13] https://github.com/AgriculturalSemantics/agro
[14] https://github.com/ApolloDev/apollo-sv
[15] http://basic-formal-ontology.org/
[16] http://www.obofoundry.org/ontology/bto.html

- Common Anatomy Reference Ontology (**CARO**)[17]
- Chemical Entities of Biological Interest (**CHEBI**)[18]
- Chemical Methods Ontology (**CHMO**)[19]
- Cell Ontology (**CL**)[20]
- Human Disease Ontology (**DOID**)[21]
- Drug Ontology (**DRON**)[22]
- Human developmental anatomy ontology (**EHDAA2**)[23]
- Environment Ontology (**ENVO**)[24]
- Food-Biomarker Ontology (**FOBI**)[25]
- FoodOn[26]
- Gazetteer (**GAZ**)[27]
- Gene Ontology (**GO**)[28]
- Human Phenotype Ontology (**HP**)[29]
- Information Artifact Ontology (**IAO**)[30]
- Mammalian Phenotype Ontology (**MP**)[31]
- NCBI organismal classification (**NCBITaxon**)[32]
- Ontology for Biomedical Investigations (**OBI**)[33]
- Phenotype And Trait Ontology (**PATO**)[34]
- Population and Community Ontology (**PCO**)[35]
- Plant Experimental Conditions Ontology (**PECO**)[36]
- Plant Ontology (**PO**)[37]
- Relation Ontology (**RO**)[38]
- Symptom Ontology (**SYMP**)[39]
- Uberon[40]

[17] https://github.com/obophenotype/caro/
[18] https://www.ebi.ac.uk/chebi/
[19] http://obofoundry.org/ontology/chmo.html
[20] https://obophenotype.github.io/cell-ontology/
[21] https://disease-ontology.org/
[22] https://github.com/ufbmi/dron
[23] http://obofoundry.org/ontology/ehdaa2.html
[24] http://environmentontology.org/
[25] http://www.obofoundry.org/ontology/fobi.html
[26] https://foodon.org/
[27] http://obofoundry.org/ontology/gaz.html
[28] http://geneontology.org/
[29] http://www.obofoundry.org/ontology/hp.html
[30] https://github.com/information-artifact-ontology/IAO/
[31] http://www.informatics.jax.org/vocab/mp_ontology/
[32] http://obofoundry.org/ontology/ncbitaxon.html
[33] http://obi-ontology.org/
[34] http://obofoundry.org/ontology/pato.html
[35] https://github.com/PopulationAndCommunityOntology/pco
[36] http://www.obofoundry.org/ontology/peco.html
[37] http://www.obofoundry.org/ontology/po.html
[38] https://oborel.github.io/
[39] http://symptomontologywiki.igs.umaryland.edu/
[40] http://uberon.github.io/

- Units of measurement ontology (**UO**)[41]
- Experimental condition ontology (**XCO**)[42]

This set of ontologies covers 4 031 out of 11 693 (34.5%) MeSH descriptors present in the corpus (271 714 out of 872 838 (31.1%) descriptor occurrences). This relatively low coverage contrasts with the very wide range of topics these ontologies cover. This is possibly due to MeSH being designed for indexing publications, and thus having terms oriented toward more general topics instead of granular concepts. For example, one of the frequently occurring MeSH descriptors is *Fishes*, an imprecise term that does not have a strict definition. The NCBITaxon ontology includes many species of fish (and their biological taxonomy), but not this particular term. Other OBO Foundry ontologies also describe specific terms very well, but lack the more general/imprecise ones.

One way to cope with this issue would be to include an additional, "meta-level" biomedical ontology. MeSH could play this role, but it has several significant drawbacks. First, being a thesaurus, it is less expressive than ontologies. Second, its non-standard structure hinders its reuse with OWL ontologies that adhere to different design principles. Ultimately, the Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT) was selected. It covers well the "general" biomedical terms and contains very expressive relations [8].

To integrate it with OBO Foundry ontologies, the terminology was converted to OWL[43] and inserted into a knowledge base. Next, connections between identical (or related) terms in OBO Foundry ontologies and SNOMED CT were created, using two sources of mappings. Some OBO ontologies already contained cross-database references to SNOMED CT – these references were normalized and converted to `skos:closeMatch` relations. Additionally, all mappings for SNOMED CT were retrieved from the BioPortal API. In total, 43 595 unique mappings were inserted into the KB, covering 37 999 of all 354 318 entities (10.7%) in SNOMED CT. From here, the combined OBO Foundry with SNOMED CT knowledge base will be referred to as **OBO/SNOMED**.

MeSH descriptor coverage was then re-evaluated. The new set of ontologies covers 7 696 out of 11 693 (65.8%) MeSH descriptors, which constitutes 569 104 out of 872 838 (65.2%) descriptor occurrences. The twenty most frequently occurring, yet uncovered MeSH descriptors include: *food contamination*, *food microbiology*, *food safety*, *pesticide residue*, and other terms directly related to food safety. This implies that the constructed ontology is still incomplete in this regard. This problem is not solvable with the immediately available resources, and solving it is out of scope of this contribution.

The knowledge graph was inspected for inconsistencies and other issues that may hamper the classification algorithm. The first discovered issue was the lack of consistency among cross-ontology references. They are supposed to connect similar/identical terms originating from different ontologies. Some such refer-

---

[41] https://github.com/bio-ontology-research-group/unit-ontology
[42] https://rgd.mcw.edu/rgdweb/ontology/view.html?acc_id=XCO:0000000
[43] https://github.com/IHTSDO/snomed-owl-toolkit

ences are specified with loose annotation properties pointing to textual identifiers, not their URIs. Some target terms did have their identifiers attached using another annotation property, but many did not. These problems were mitigated with SPARQL UPDATE queries that transformed OBO references into traversable `skos:closeMatch` properties. Note that these issues are known and their detailed discussion can be found in the work of Laadhar et al. [16]. Other issues, e.g. triples with properties pointing to erroneous URIs, object properties referring to literal values, and annotation properties referring to objects, were also fixed with SPARQL queries. Such errors would decrease the accuracy of the method, due to it relying on the ontology for domain knowledge.

At the end, the OBO/SNOMED graph became relatively large, containing almost 49 million triples. Handling such a large dataset presents significant challenges. Therefore, the graph was inspected for triples of little value to text-based information retrieval. Unnecessary information was removed including provenance information, metadata, links to external databases, and more. This resulted in a graph with approximately 25 million triples.

5   Proposed approach

Let us now describe the method used to perform topical publication classification. Proposed approach is based on the following assumptions. (1) Classification should be as accurate as possible. (2) Solution should be fast and scalable, as target ontologies, depending on the discipline, can have tens of millions of triples. (3) The method should robustly cope with larger-than-memory datasets, without significant performance degradation. (4) Solution should be easily applicable to any domain. This implies reliance on the information contained in the KB, while "domain-specific code" should be limited to a minimum.

Figure 1 presents an overview of the method. First, using named entity recognition (NER), mentions are identified in the text (Sect. 5.1). Then, for each mention, a set of candidate entities is produced, based on a full-text query to a search engine (Sect. 5.2). Neighborhoods of candidates are taken into account, using an algorithm that exploits the semantic meaning of relations between entities (Sect. 5.3). Candidates are compared to the mention, and each is assigned a similarity score (Sect. 5.4). Additionally, connections between candidates are used to discard the least coherent choices (Sect. 5.5). Finally, the list of entities is enhanced using information from the KB, and the final set of terms most relevant to the publication is returned (Sect. 5.6). Let us now describe each operation in more detail.
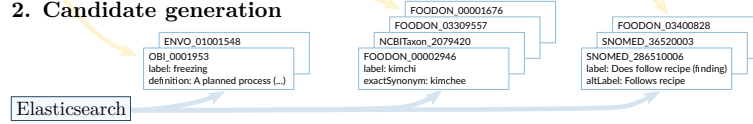
5.1   Named entity recognition

The first stage of the method identifies mentions, in publication abstracts that may correspond to "entities of interest". In later stages, these mentions will be linked to the entities in the knowledge base. There are several ready solutions to this problem in the biomedical domain, such as the scispaCy biomedical pipeline
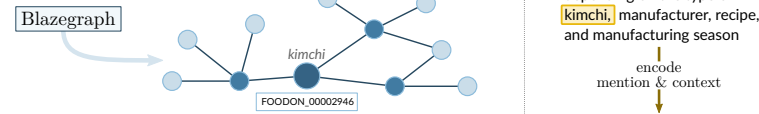
1. Named entity recognition

The supercooling degree (SD), which refers to the difference between the ice nucleation temperature and freezing point of kimchi, varies depending on the type of kimchi, manufacturer, recipe, and manufacturing season.
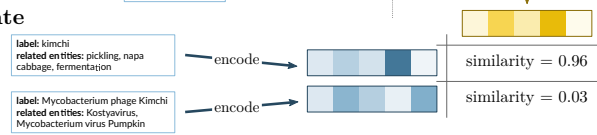
2. Candidate generation

FOODON_00001676
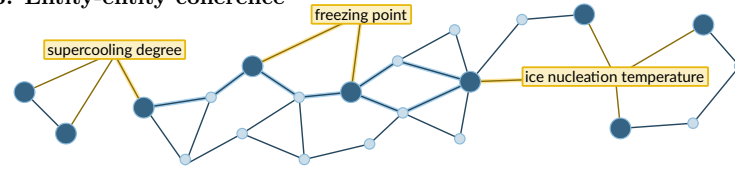FOODON_03309557
NCBITaxon_2079420
FOODON_00002946
label: kimchi
exactSynonym: kimchee

ENVO_01001548
OBI_0001953
label: freezing
definition: A planned process (...)

FOODON_03400828
SNOMED_36520003
SNOMED_286510006
label: Does follow recipe (finding)
altLabel: Follows recipe

Elasticsearch

3. Graph expansion

Blazegraph

kimchi

FOODON_00002946

depending on the type of kimchi, manufacturer, recipe, and manufacturing season

encode
mention & context

4. Candidate ranking

label: kimchi
related entities: pickling, napa cabbage, fermentation

encode

similarity = 0.96

label: Mycobacterium phage Kimchi
related entities: Kostyavirus, Mycobacterium virus Pumpkin

encode

similarity = 0.03

5. Entity-entity coherence

freezing point

supercooling degree

ice nucleation temperature

6. Candidate selection and results enhancement

supercooling degree

kimchi

kimchi

freezing point

parent entities

explicit mentions

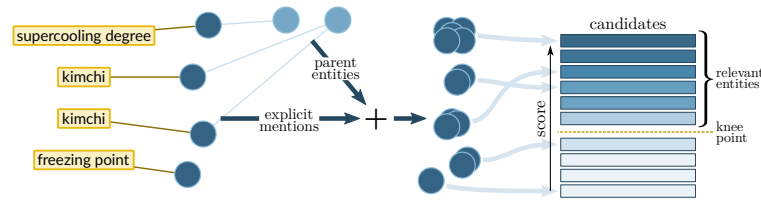candidates

score

relevant entities

knee point

Fig. 1. Overview of the method

(`en_core_sci_lg`), which achieves a 69% F_1 score in NER, on the very demanding MedMentions dataset [21] and it was chosen to be used here. The final set of mentions is additionally extended with explicitly provided keywords, which are available for the EFSA corpus.

5.2   Candidate generation

Selecting candidate entities for a mention is not trivial when the number of entities in the KB is large; e.g. OBO/SNOMED is too large to be loaded into memory of a standard computer. Thus, Elasticsearch[44] was used for candidate

---

[44] https://www.elastic.co/what-is/elasticsearch

generation. Specifically, an Elasticsearch index was constructed, fields of which depended on types of text properties present within the KB. The unique identifier for each record in the index is simply the entity's URI. For each type of text property, a numeric weight was assigned for use when searching, to indicate how relevant the field is expected to be. Additionally, several fields are included for caching results of compute-intensive operations, described in detail in Sects. 5.3 and 5.6. Finally, the constructed index for the OBO/SNOMED knowledge base consists of 3 739 476 valid entity linking targets.

5.3   Graph expansion

A candidate entity does not provide context information by itself – it is necessary to look at related entities to obtain a better sense of what it describes. This "graph expansion" approach was used successfully in a textual entailment system [29]. Before attempting to design an algorithm for this problem, KB was examined and several observations were made.

O1: Traversing the graph only along the edges' original directionality may be too limiting. For example, with the `rdfs:subClassOf` relations this would result in only traversing the class hierarchy upward. Thus, the algorithm should traverse edges in both directions.

O2: Processing large graphs with long-tail distributions of node degrees is challenging [11]. Encountering a vertex with a large number of neighbors forces the algorithm to visit all of them. Moreover, the relative amount of information a neighbor provides can vary considerably. Within large KBs one can find entities with numerous connections (e.g. many instances of one class), which makes the meaning of such an entity intuitively less "concentrated". Conversely, an entity with fewer connections is more likely to have a more concrete and relevant meaning.

O3: Similarly, if an entity has many connections of type A, but very few of type B, the B-type edges are likely provide more "focused" information. For instance, the *analgesic* class from CHEBI is connected to 65 substances that can be used as analgesics. When expanding from the *analgesic* entity, they may introduce "noise". On the other hand, connections such as *analgesic is a subclass of drug* do provide important contextual information.

O4: Finally, different types of properties have different meanings and can be intuitively translated to connections of varying importance. For example, the `skos:closeMatch` relation implies greater similarity than `rdfs:subClassOf` or `obo:RO_0002604` (is opposite of).

Based on these observations, the following formula for calculating edge weights (w_{rel}) was proposed (3).

$$l(o) = |(s', p', o) \in \text{KB} \cup (o, p', o') \in \text{KB}| \tag{1}$$

$$g(s, p, d) = \begin{cases} |(s, p, o') \in \text{KB}| & \text{if } d = \text{'spo'} \\ |(s', p, s) \in \text{KB}| & \text{if } d = \text{'ops'} \end{cases} \tag{2}$$

$$\mathrm{w}_{\mathrm{rel}}(s, p, o, d) = w_p \, \mathrm{l}(o)^{f_l} \, \mathrm{g}(s, p, d)^{f_g} \tag{3}$$

Here s, p, o (subject, predicate, object) is a triple from the KB that may be inverted to allow traversing the graph in either direction (**O1**). In such case, the d parameter is set to *ops* to indicate the inversion, otherwise it is *spo*. The final weight consists of three components:

- w_p – the constant base weight for property p (**O4**). The base weights have to be picked manually, based on the user's expertise, ontology documentation, and the use of the property in practice. The value typically ranges from 0.1 for very close, identity-like ties to 4, for the loose connections.
- $\mathrm{l}(o)$ – the total number of links to and from the object of the triple (**O2**). By considering only the number of links of the object, the algorithm discourages visiting nodes with high cardinality, while leaving the node is not penalized. This is additionally scaled by a tunable parameter, f_l (0.5 by default).
- $\mathrm{g}(s, p, d)$ – the number of links from subject s, through predicate p, in direction d (**O3**). This is also scaled by a parameter, f_g (0.5 by default).

In the equations, s', p', o' refer to (respectively) any subject, predicate, and object matching the given triple pattern. Expressions in the form of (s, p, o) indicate triple patterns present in the KB.

To compute $\mathrm{w}_{\mathrm{rel}}$ effectively, the following approach is used. First, for each node x in the KB, $\mathrm{l}(x)$ is calculated and the result inserted back into the graph. Next, a list of all (s, p, o) triples in the KB is generated, along with their $\mathrm{l}(s)$ and $\mathrm{l}(o)$. The list is enhanced with inverted triples, to enable traversing edges in both directions. This results in a list that may be much larger than the system memory available, which necessitates careful processing of the data (in parts). To be able to perform grouping operations effectively, the list is sorted on disk and repartitioned by s, producing partitions that can be safely grouped in isolation. Then, it is grouped by s, p, and d, which corresponds to the $\mathrm{g}(s, p, d)$ function. Computing $\mathrm{w}_{\mathrm{rel}}$ is straightforward now, and is done by iterating over all obtained groups. For each group only c_{max} (4 by default) connections with lowest weights are kept. The partitioned data is processed in parallel using the Dask Python library [6].

The algorithm outputs a list of triples with weights that when inserted into the KB yield a weighted, traversable graph. A weighted graph can be more generally framed as an attribute graph, where any edge between two nodes (a triple) can be annotated with additional information. Blazegraph implements a solution to this called Reification Done Right (RDR), in which any triple can be either the object or the subject of another triple [12].

Perhaps the most obvious solution to finding neighboring entities in the knowledge graph would be to employ SPARQL queries, however, this would be hardly a typical use case for this language. A better fit would be a Gather-Apply-Scatter (GAS) algorithm, similar to the PowerGraph approach [11]. GAS allows one to write high-performance, parallel graph analytics algorithms. Blazegraph implements this approach[45] via a Java API. The graph expansion algorithm

---

[45] https://github.com/blazegraph/database/wiki/RDF_GAS_API

was implemented using this method and compiled as a Blazegraph plugin. The algorithm obtains edge weights of triples using RDR. The code is a modified version of the breadth-first search algorithm, taking into account maximum allowed traversal depth and maximum distance of found neighbors. The search is executed in parallel, across multiple threads.

As the neighborhood of an entity depends solely on the ontology, and not the particular search being made, it is possible to perform graph expansion for each possible candidate beforehand, to speed up classification. Then, the neighboring entities are saved to the Elasticsearch index, along with the distances to them (in the `related_entities` and `related_entities_weights` fields). This removes the need to run and query Blazegraph during runtime, improving classification speed and reducing the overall memory footprint.

5.4   Candidate ranking

Obtained candidate entities are then rated by their relevance to the mention. The algorithm should consider both the context in which the mention appears, as well as the neighborhood of the candidate entity within the KB. The proposed solution uses two types of text vector representations – semantic and lexical. The semantic component employs word embeddings provided by the scispaCy library. While word embeddings have superior performance to lexical-based methods, one must also consider that out-of-vocabulary words will not produce a vector and thus cannot be compared. Here, for scientific texts, it is unreasonable to expect the embedding model to cover all possible words, some of which may be highly specialized and/or rarely occurring. Hence, the lexical component is introduced, based purely on character-level similarity between text fragments. To construct a lexical vector, the text fragment is split into 3-4 character n-grams. Next, the unique n-grams are grouped by their hashes and counted.

To compute the score of a candidate, its and all of its neighbors' text properties are vectorized. This produces two matrices, $L_{n \times m_l}$ and $S_{n \times m_s}$, for lexical and semantic vectors, respectively. Here n is the number of all text properties from all entities that are taken into account. The number of columns depends on the length of the vectors. The matrices are L2-normalized row-wise, with the exception of zero-length vectors. Additionally, a vector of distances to related entities is retrieved (w_e), with values starting from zero (for the candidate entity itself). As described in Sect. 5.2, text properties can have varying importance, which is considered here in the form of w_p, a vector of property weights. Finally, the similarity between the mention and other text fragments is calculated in bulk using (4).

$$d = w_p \frac{w_l \, \mathrm{a}\,(Ll) + w_{sm} \, \mathrm{a}\,(Ss_m) + w_{sc} \, \mathrm{a}\,(Ss_c)}{1 + w_e} \tag{4}$$

$$\mathrm{a}(x) = (1 + \exp\,(\alpha - \beta x))^{-2} \tag{5}$$

Here l, s_m, s_c refer to the lexical vector of the mention, its semantic vector, and the semantic vector of the sentence the mention is in (its context), respectively. It is assumed that these vectors are L2-normalized. Moreover, w_l, w_{sm},

w_{sc} are tunable weights for the three components. Next, raw cosine similarity values are put through the activation function a, presented in (5). It is a case of the generalized logistic function. Its purpose is to discard low similarity scores and "promote" good matches. The α and β parameters can be tuned to control its steepness and translation along the X axis. The function will return values from the $(0, 1)$ interval. Equation (4) produces an n-length vector of similarity scores between the mention and each of the considered text fragments. The score is accumulated by simply summing the elements of the vector d. The actual code for calculating this formula uses several performance optimizations, such as a lookup table for the activation function. Additionally, numerically-heavy routines were compiled using the Numba JIT compiler [17].

To further speed up classification, the feature vectors for text from the KB are precomputed during indexing. They are then stored and retrieved during classification, minimizing the amount of text processing at runtime. The cached vectors can take up large amounts of memory. For the OBO/SNOMED KB problems occurred even on a computer with $\sim$32 GB of memory, and for other KBs required memory may increase further. This is solved by storing the vectors using the Lightning Memory-Mapped Database (LMDB)[46], which offers very high random-read performance and supports concurrent reads. This approach, however, necessitates the use of an SSD as the backing storage medium, as traditional hard drives are unsuitable for rapid random reads.

As a result of this step, for each mention, an ordered list of candidates with assigned scores is obtained. The greater the score, the more likely the candidate is to be a good match for the mention.

5.5   Entity-entity coherence

To help disambiguate named entities, the entity-entity coherence algorithm, introduced by Hoffart et al. [13] is used, albeit with several modifications – most notably the preprocessing stage is different. Original entity-entity coherence values are based on Wikipedia article similarity, which produces a dense similarity matrix. This would be hard to implement efficiently here, given the high complexity of algorithms computing all-pairs distance matrices in graphs.

Therefore, first, the highest scoring candidate entities across all mentions are selected, to discard noisy low-confidence scores. A sparse similarity matrix is constructed, which represents how closely candidates are related to each other. This is established by effectively approximating the shortest path between each pair of the candidates, using their cached neighborhoods from the graph expansion step. Resulting matrix will not be very dense. However, the most important connections should be present. Finally, the set of candidates is narrowed down by greedily removing the least strongly connected candidates. The algorithm outputs a mapping of candidate entities to boost values, scaled to be greater or equal to 1. This boost is used to multiply the scores of those entities, possibly

---

[46] https://symas.com/lmdb/

changing the ordering of candidates for each mention. The candidate ranking lists are then sorted again, to account for this change.

To enhance the performance of the method, several additional "bookkeeping" data structures and optimizations are introduced. Additionally, the last stage of the algorithm, being heavy in numerical computations, is compiled using the Numba JIT compiler.

5.6   Candidate selection and results enhancement

Having identified and ranked candidate entities for each mention, the final step is to determine the most relevant terms that describe the publication. The classification result should also include concepts that are not explicitly mentioned in the text. This is achieved by using a technique similar to the one employed in the CSO Classifier [22], where the final set of terms is enhanced with "parent" entities, higher up in the "semantic hierarchy". Depending on the ontology this may translate to, for example, a subclassing relationship or the `skos:broader` property. The list of parent entities is established using a SPARQL query with parameters, which allow for specifying parent properties or inverse parent properties, to accommodate both approaches to expressing such relationships. Additionally, each parent is given a weight $w_p = links^{-\alpha}$, where $\alpha \in (0, 1)$. Here $links$ is the total number of links the parent has and α is a scaling factor, set to 0.3 by default. These parent entities along with their weights are saved to Elasticsearch during indexing, in the `parent_entities` and `parent_entities_weights` fields. This information is later retrieved during classification.

Similarly to the CSO Classifier, the ranking algorithm aggregates the top-scoring candidates across mentions, increasing their score based on how frequently they were identified in text and with what diversity (number of unique mention lemmas). From the final list of candidates only a portion is kept using, which is by finding the "knee point" in the distribution of scores. The knee is established using the Kneedle algorithm, which can adjust to different situations robustly [24] and is shown to perform significantly better than any fixed cutoff value.

5.7   Summary

The presented method makes extensive use of preprocessing to offload most of the computationally intensive tasks to run-only-once jobs and in turn, speed-up classification. Part of the data is cached in Elasticsearch, which is accessed over an HTTP API, with only two batch requests per classified document. The encoded text vectors' significant size would make transporting them over HTTP inefficient. Thus, they are stored in an embedded database (LMDB), which achieves zero-copy reads, owing to its memory-mapped architecture.

Adaptability to other domains is achieved through the extensive use of configuration settings and modular design. One can easily create new mention detectors, semantic vector encoders, and lexical vector encoders, with a few lines of

Python code. Other aspects of the classifier, such as the various tunable parameters of algorithms presented above, can be changed using a YAML configuration file. The process of retooling the entire pipeline to an entirely new domain is largely reduced to changing the configuration file and providing an appropriate ontology.

6   Experimental results

A thorough quantitative classification accuracy evaluation of the classifier could not be performed, due to the lack of a labeled dataset. Thus, the scope of the assessment was limited to identifying obvious misclassifications that would negatively impact the precision metric. For this purpose, articles from the EFSA and PubMed datasets were classified and selected results examined manually. Table 2 presents example output of the method, with two identified errors highlighted in bold.

It can be observed that the method performs generally well, albeit it tends to frequently return terms that do not seem to have much importance, such as: *species*, *application*, *food*, *population*. Usually, it is able to perceive context, but can still fail in some cases when understanding of the entire sentence is needed for correctly identifying the mention. Unfortunately, this is hard to achieve with simple word embeddings. However, rarely occurring and very specific terms such as taxa, geographic locations, and chemicals are identified easily and seemingly with good accuracy.

The performance of the method was measured on a modern Linux workstation with a 3.6 GHz 6-core CPU, 32 GB of system memory and an NVMe SSD rated at 480 000 IOPS. From the EFSA and PubMed corpora 50 publications were selected randomly and classified in batch. Only the wall clock time taken by classification was measured, excluding the time required to load the code and the models into memory. The experiment was repeated ten times. Classifying the entire EFSA batch took 78.48 s in average, with the mean time to classify a publication at 1.57 s. For PubMed, classifying the batch took 80.49 s on average.

To examine possible bottlenecks, the classification of a batch of 50 EFSA publications was profiled using cProfile. The results are presented in Fig. 2, with only the most important calls included.

Over 60% of the program's execution time is spent generating candidates, which seems counter-intuitive, as this should be simply a matter of querying Elasticsearch. Querying is indeed present here (taking up only 3.4% of the total time), but for efficiency reasons several other operations are performed as well. LMDB is queried for the stored lexical and semantic vectors (11.9%). These vectors have to be then decoded with the pickle protocol to Python objects (11.5%). Finally, the found candidates and their neighbors are rearranged into data structures optimized for the later stages (33.8%).

This last step can be especially confusing, raising question why such seemingly simple operation requires so much time. The main reason is that code compiled with Numba, in later stages, requires its input data structures to be

Table 2. Example classification of a food safety PubMed article [7]

URI	Label	Mention lemmas	Score
obo:CHEBI_26208	polyunsaturated fatty acid	blood fatty acid, plasma fatty acid, fatty acid profile	82.88
obo:NCBITaxon_29073	Ursus maritimus	polar bear, polar bear intake	43.93
obo:CHEBI_35366	fatty acid	fatty acid	33.97
obo:ENVO_02500036	environmental pollution	pollution	33.86
obo:GAZ_00052766[1]	Illoqqortoormiut	East Greenland, Ittoqqortoormiit	32.30
obo:CHEBI_5692	hexachlorobenzene	hexachlorobenzene	31.69
obo:OBI_0000181	population	population	29.20
obo:GAZ_00001507	Greenland	Greenland	27.75
obo:NCBITaxon_9709	Phocidae	seal	25.13
snomedct:19314006	Seal (organism)	seal	22.48
obo:FOODON_03411343	whale	whale	22.45
snomedct:733446001[2]	**Canadian (ethnic group)**	**canadian**	22.18
snomedct:226365003	N-3 fatty acid (substance)	n-3 fatty acid	19.81
obo:CHMO_0002820	concentration	concentration	19.73
obo:CHEBI_34852	mirex	mirex	17.69
obo:CHEBI_61204	docosapentaenoic acid	docosapentaenoic acid	17.29
obo:CHEBI_34623	chlordane	chlordane	16.85
obo:CHEBI_27208	unsaturated fatty acid	blood fatty acid, plasma fatty acid, fatty acid profile	16.62
snomedct:223503004[3]	**North America (geographic location)**	**North**	16.27
obo:PATO_0000025	composition	composition	14.90
obo:UBERON_0001969	blood plasma	plasma	14.72
obo:FOODON_03412406	bear	polar bear, polar bear intake	14.49
obo:GAZ_00228284	Northwest Atlantic Ocean coastal waters of Greenland	Greenland	13.87
5 more results omitted			

[1] Illoqqortoormiut is an alternative name of Ittoqqortoormiit. The classification is correct.

[2] Misclassification: the text refers to the "Canadian guideline levels", not the ethnic group of Canadians.

[3] Misclassification: the text refers to Northern Greenland (the North), not the North America. Although Greenland is in fact considered to be a part of North America, this "technically correct" entry is probably just the result of coincidence.
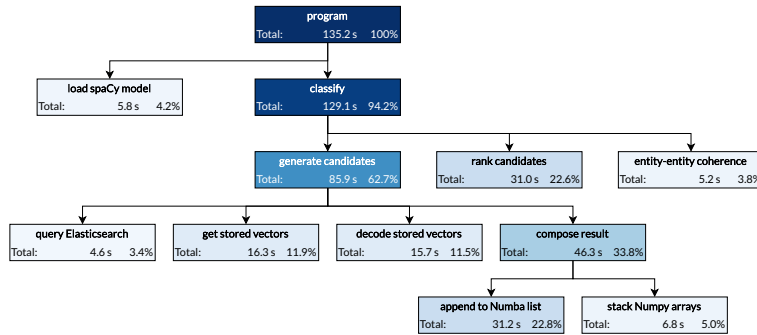
Fig. 2. Profiler diagram for classifying 50 EFSA publications in a batch

Numba-compatible. This necessitates "boxing" of Python objects to be performed at some point, which incurs a performance hit. Another option would be not to use Numba, but that would make the program even slower. Another compute-intensive step is candidate ranking, where most of the time is spent in optimized numerical routines.

7   Conclusions

The proposed method is an adaptable, scalable, and performant solution to the topical publication classification problem, easily handling KBs containing millions of entities. Classification results seem satisfactory, although a detailed evaluation remains to be performed on a labeled dataset. The method is robust, modular base which can be iteratively improved, to achieve better accuracy. In particular, further research into candidate generation and how the algorithm perceives the context of both the abstract and the knowledge base is needed.

Regarding classification speed, substantial improvements could be made by parallelizing the code. However, currently this could not be easily done, due to the Python Global Interpreter Lock (GIL), which prevents the simultaneous execution of multiple Python interpreter threads [19]. There is also a significant amount of overhead involved at several steps, necessitated by transitioning from interpreted to native code. The only reasonable solution to these problems would be to drop pure Python as the implementation language altogether in favor of C++ or Cython, which will be investigated in future research.

Additional materials and data related to this work, as well as the most important algorithms mentioned above, were published on GitHub[47].

---

[47] https://github.com/Ostrzyciel/food-safety-classif

Bibliography

[1] Arbabi, A., Adams, D.R., Fidler, S., Brudno, M.: Identifying clinical terms in medical text using ontology-guided machine learning. JMIR medical informatics **7**(2), e12,596 (2019)

[2] Aromataris, E., Riitano, D.: Constructing a search strategy and searching for evidence. American Journal of Nursing **114**(5), 49–56 (2014)

[3] Baba, V.V., HakemZadeh, F.: Toward a theory of evidence based decision making. Management decision (2012). DOI 10.1108/00251741211227546

[4] Boqvist, S., Söderqvist, K., Vågsholm, I.: Food safety challenges and One Health within Europe. Acta Veterinaria Scandinavica **60**(1), 1–13 (2018)

[5] Cock, P.J., Antao, T., Chang, J.T., Chapman, B.A., Cox, C.J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., et al.: Biopython: freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics **25**(11), 1422–1423 (2009)

[6] Dask Development Team: Dask: Library for dynamic task scheduling (2016). URL https://dask.org

[7] Deutch, B., Pedersen, H., Hansen, J.: Dietary composition in Greenland 2000, plasma fatty acids and persistent organic pollutants. The Science of the total environment **331 1-3**, 177–88 (2004)

[8] Donnelly, K., et al.: SNOMED-CT: The advanced terminology and coding system for eHealth. Studies in health technology and informatics **121**, 279 (2006)

[9] Dooley, D.M., Griffiths, E.J., Gosal, G.S., Buttigieg, P.L., Hoehndorf, R., Lange, M.C., Schriml, L.M., Brinkman, F.S., Hsiao, W.W.: FoodOn: a harmonized food ontology to increase global food traceability, quality control and data integration. npj Science of Food **2**(1), 1–10 (2018)

[10] Fraser, K.C., Nejadgholi, I., De Bruijn, B., Li, M., LaPlante, A., Abidine, K.Z.E.: Extracting UMLS concepts from medical text using general and domain-specific deep learning models. arXiv preprint arXiv:1910.01274 (2019)

[11] Gonzalez, J.E., Low, Y., Gu, H., Bickson, D., Guestrin, C.: Powergraph: Distributed graph-parallel computation on natural graphs. In: 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12), pp. 17–30 (2012)

[12] Hartig, O., Thompson, B.: Foundations of an alternative approach to reification in RDF. arXiv preprint arXiv:1406.3399 (2014)

[13] Hoffart, J., Yosef, M.A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., Weikum, G.: Robust disambiguation of named entities in text. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 782–792 (2011)

[14] Jonnalagadda, S.R., Goyal, P., Huffman, M.D.: Automating data extraction in systematic reviews: a systematic review. Systematic reviews **4**(1), 78 (2015)

[15] Jonquet, C., Shah, N., Youn, C., Callendar, C., Storey, M.A., Musen, M.: NCBO Annotator: semantic annotation of biomedical data. In: International Semantic Web Conference, Poster and Demo session, vol. 110 (2009)

[16] Laadhar, A., Abrahão, E., Jonquet, C.: Investigating one million XRefs in thirthy ontologies from the OBO world. In: 11th International Conference on Biomedical Ontologies (ICBO) (2020)

[17] Lam, S.K., Pitrou, A., Seibert, S.: Numba: a LLVM-based Python JIT compiler. In: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, pp. 1–6 (2015)

[18] Li, H., Chen, Q., Tang, B., Wang, X., Xu, H., Wang, B., Huang, D.: CNN-based ranking for biomedical entity normalization. BMC bioinformatics **18**(11), 79–86 (2017)

[19] Meier, R., Gross, T.: Reflections on the compatibility, performance, and scalability of parallel Python. Proceedings of the 15th ACM SIGPLAN International Symposium on Dynamic Languages (2019)

[20] Morton, S., Berg, A., Levit, L., Eden, J., et al.: Finding what works in health care: standards for systematic reviews. National Academies Press (2011)

[21] Neumann, M., King, D., Beltagy, I., Ammar, W.: ScispaCy: Fast and robust models for biomedical natural language processing. In: Proceedings of the 18th BioNLP Workshop and Shared Task, pp. 319–327. Association for Computational Linguistics, Florence, Italy (2019). DOI 10.18653/v1/W19-5034. URL https://www.aclweb.org/anthology/W19-5034

[22] Salatino, A.A., Osborne, F., Thanapalasingam, T., Motta, E.: The CSO classifier: Ontology-driven detection of research topics in scholarly articles. In: International Conference on Theory and Practice of Digital Libraries, pp. 296–311. Springer (2019)

[23] Salatino, A.A., Thanapalasingam, T., Mannocci, A., Osborne, F., Motta, E.: The computer science ontology: a large-scale taxonomy of research areas. In: International Semantic Web Conference, pp. 187–205. Springer (2018)

[24] Satopää, V., Albrecht, J.R., Irwin, D.E., Raghavan, B.: Finding a "kneedle" in a haystack: Detecting knee points in system behavior. 2011 31st International Conference on Distributed Computing Systems Workshops pp. 166–171 (2011)

[25] Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., Goldberg, L.J., Eilbeck, K., Ireland, A., Mungall, C.J., et al.: The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. Nature biotechnology **25**(11), 1251–1255 (2007)

[26] Tsafnat, G., Glasziou, P., Choong, M.K., Dunn, A., Galgani, F., Coiera, E.: Systematic review automation technologies. Systematic reviews **3**(1), 74 (2014)

[27] Uyttendaele, M., De Boeck, E., Jacxsens, L.: Challenges in food safety as part of food security: lessons learnt on food safety in a globalized world. Procedia food science **6**, 16–22 (2016)

[28] Whetzel, P.L., Noy, N.F., Shah, N.H., Alexander, P.R., Nyulas, C., Tudorache, T., Musen, M.A.: BioPortal: enhanced functionality via new web services from the National Center for Biomedical Ontology to access and use ontologies in software applications. Nucleic acids research **39**(suppl_2), W541–W545 (2011)

[29] Yadav, S., Pallagani, V., Sheth, A.: Medical knowledge-enriched textual entailment framework. In: Proceedings of the 28th International Conference on Computational Linguistics, pp. 1795–1801 (2020)

[30] Zheng, J.G., Howsmon, D., Zhang, B., Hahn, J., McGuinness, D., Hendler, J., Ji, H.: Entity linking for biomedical literature. BMC medical informatics and decision making **15**(S1), S4 (2015)