Keyword Extraction in Scientific Documents

Susie Xi Rao[1,*], Piriyakorn Piriyatamwong[1,*], Parijat Ghoshal[2,*], Sara Nasirian[3], Sandra Mitrović[3], Emmanuel de Salis[4], Michael Wechner[5], Vanya Brucker[5], Peter Egger[1] and Ce Zhang[1]

[1]Chair of Applied Economics, ETH Zurich, Switzerland
[2]Neue Zürcher Zeitung AG, Zurich, Switzerland
[3]Dalle Molle Institute for Artificial Intelligence, Lugano, Switzerland
[4]Haute-Ecole Arc, Neuchâtel, Switzerland
[5]Wyona AG, Zurich, Switzerland

Abstract
The scientific publication output grows exponentially. Therefore, it is increasingly challenging to keep track of trends and changes. Understanding scientific documents is an important step in downstream tasks such as knowledge graph building, text mining, and discipline classification. In this workshop, we provide a better understanding of keyword and keyphrase extraction from the abstract of scientific publications.

1. Introduction

Keyphrases are single- or multi-word expressions (often nouns) that capture the main ideas of a given text, but do not necessarily appear in the text itself [1, 2, 3]. Keyphrases have been shown to be useful for many tasks in the Natural Language Processing (NLP) domain, such as (1.) indexing, archiving and pinpointing information in the Information Retrieval (IR) domain [3, 4, 5, 6], (2.) document clustering [3, 7, 8], and (3.) summarizing texts [3, 9, 10, 11], just to name a few.

Keyphrase extraction has been at the forefront of various application domains, ranging from the scientific community [1, 2, 12], finance [13, 14], law [15], news media [11, 16, 17], patenting [18, 19], and medicine [20, 21, 22]. Despite being a seemingly straightforward task for human domain experts, performing automatic keyphrase extraction is a challenging task.

Challenge 1: Benchmark Dataset and Keyword Reference List. One main reason is the lack of benchmark datasets and keyword reference lists, as authors often do not provide their keyphrase list unless explicitly requested or required to do so [3]. In scientific publications, we see a large variation across domains (e.g., economics, computer science, mathematics, engineering fields, humanities). For instance, publications in some disciplines, such as economics, are required to have author-generated or journal-curated keywords, while in other domains, such as computer science and engineering fields, not all publication venues (e.g., journals, proceedings) require authors to input keywords.

In less technical domains, such as news media, keyphrase lists may be more accessible in terms of the availability and the ease of manually curating the keyphrase list, even when reference lists are not readily available. This is because in the news domain, people have particular interests in Named Entities (labelled entities such as person, location, event, time), as we will discuss in Section 6. However, manually curating the keyphrase list in general is often practically infeasible–hiring domain experts is costly, while crowdsourcing the annotation is difficult to control the quality [2, 3, 11].

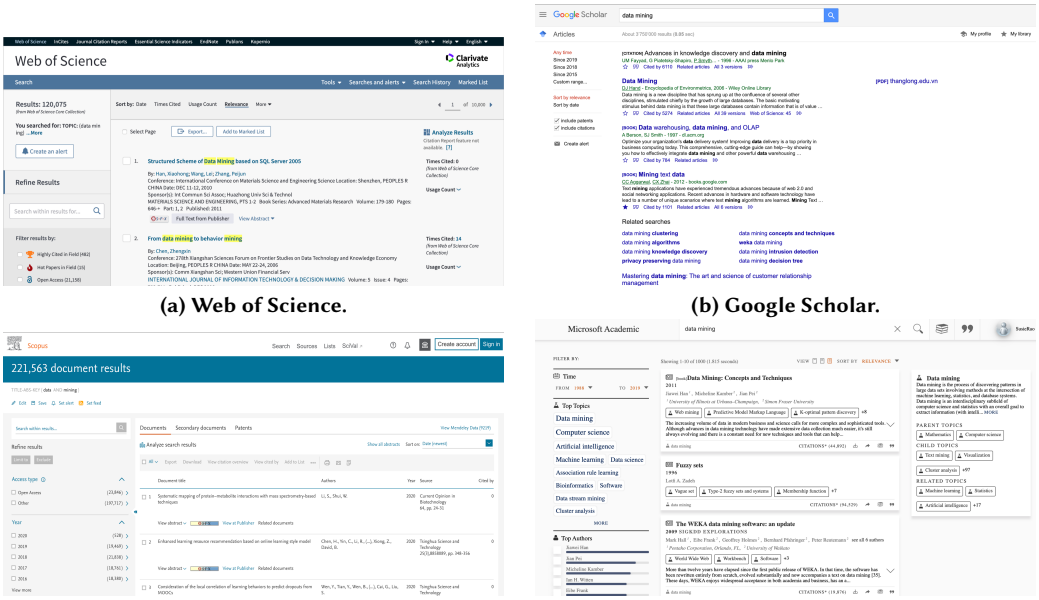
With limited availability of benchmark datasets, large language models–which succeed in other NLP tasks–simply fail to optimize and generalize, as they generally require a large, well-annotated training dataset [16]. The lack of training datasets also poses challenges for the evaluation of keyword extraction systems.

Challenge 2: Evaluation of Keyword Extraction. Defining an evaluation protocol and a corresponding metric is far from trivial for the following reasons.

(1.) We should look at the ground truth list of keywords in a critical way. As we mentioned above, there can exist more than one ground truth list of keyphrases given an abstract. The keyword list provided in our dataset is a reference list of words provided by authors or by publishers. One should only treat

SwissText 2022: Swiss Text Analytics Conference, June 08–10, 2022, Lugano, Switzerland
*Corresponding author.
†These authors contributed equally.
srao@ethz.ch (S. X. Rao); ppiriyata@ethz.ch (P. Piriyatamwong); parijat.ghoshal@nzz.ch (P. Ghoshal); sara.nasirian@supsi.ch (S. Nasirian); sandra.mitrovic@supsi.ch (S. Mitrović); emmanuel.desalis@he-arc.ch (E. d. Salis); michael.wechner@wyona.com (M. Wechner); vanya.brucker@wyona.com (V. Brucker); pegger@ethz.ch (P. Egger); cezhang@ethz.ch (C. Zhang)

CEUR Workshop Proceedings (CEUR-WS.org)



(a) Web of Science. **(b) Google Scholar.**

(c) Scopus. **(d) Microsoft Academic.**

Figure 1: Comparison of various academic products with the query for "data mining".

this list as a reference list, but not the one and only correct list of keywords.

(2.) There are different aims in extracting keyphrases in system design. As we will introduce in the rationale of designing the three systems in Section 3, the systems are designed to tackle various problems and, therefore, are optimized for different use cases. System 1 uses a simple TextRank algorithm (see Section 4), which outputs the most prominent set of keyphrases/keywords; System 2 uses TextRank on top of a clustering algorithm (see Section 5), which is targeted at grouping similar articles and then learns from the cluster of articles; and System 3 uses pre-trained models and tools on Named-Entity Recognition (NER) (see Section 6), with a goal to fully utilize existing models and tools by only pre-processing the input and/or post-processing the output.

(3.) There are different objective functions that we want to optimize. Precision, recall, accuracy, false positive rate, and false negative rate are among the most common performance metrics for various application scenarios [23]. We might also consider the order of keyphrases, for example, as sorted by criteria such as frequency, TextRank score [24, 25]. In search engines, the hit rate is also an important metric [26]. Furthermore, one can evaluate exact matches and fuzzy matches. Fuzzy matches can also be broken down into two types: "partial" matches and semantically equivalent matches [27, 28, 24]. There are other evaluation methods which account for the ranks and orders in the extracted keywords, see this Medium article for inspiration [24].

Challenge 3: Growing Number of Scientific Publications. During the last decades, the number of scientific publications has increased exponentially each year [29], making it increasingly challenging for researchers to keep track of trends and changes, even strictly in their own field of interest [3, 30]. This bolsters the need for automatic keyword extraction for the use case as a text recommendation and summarization system. The effect of increasing publications is clearly visible in major academic search engines such as Google Scholar, Web of Science, Scopus, and Microsoft Academics. In a simple query ("data mining"), three out of four failed to bring up relevant scientific publications that are prominent in the field and anticipated by human domain experts.

See the query results in Figure 1 of a keyword search "data mining" in different academic products. We can see that the search results in different products vary largely, and it could be difficult for readers to choose between the different results without having prior knowledge of the field. So far, only Microsoft Academic Services (Figure 1 (d)) has returned relevant research results that point to the most influential author and work in the field of data mining. This is because Microsoft Academic Service has enabled a hierarchical discipline classification (indexed

by keyphrases) that supports its users when reviewing the search results. In summary, without relevant and correct keyphrases, effective indexing and thus querying is not feasible.

Challenge 4: Domain-Specific Keyword Extraction. Another challenge in keyphrase extraction is its domain-specific nature. One case is when a keyphrase extractor trained in generic texts may miss out technical terms that do not look like usual keyword noun chunks, such as the chemical name "C4H*Cl" [31]. The issue arises from the tokenization step: a non-alphabetic character such as "4" and "*" might be treated as a separator, and thus such a keyword gets split into "C", "H" and "Cl", losing its original notion. Even if the separator works perfectly, this type of chemical name would still confuse keyphrase extractors that filter candidate keyphrase based on Part-of-Speech (POS) tags. This is because for POS-based extractors, it is unclear whether "C4H*Cl" is an adjective, a noun or other POS tags.

Another case is when the keyphrase consists of a mix of generic and specific words, such as "Milky Way". "Way" is generally a stopword [32], so the keyphrase extractor might only be able to detect "Milky" and throw away "Way" without realizing that the term "Way" is not a stopword in this specific context.

Finally, we would like to mention KeyBERT, a state-of-the-art BERT-based keyword extractor [33]. KeyBERT works by extracting multi-word chunks whose vector embeddings are most similar to the original sentence. Without considering the syntactic structure of the text, KeyBERT sometimes outputs keyphrases that are incorrectly trimmed, such as "algorithm analyzes", "learning machine learning". This problem only worsens with the aforementioned examples from chemistry and astronomy, since it is not straightforward how to tokenize, i.e., "split", words and how to handle non-alphabetic characters.

Our Goals and Contributions in this Workshop. Despite the challenges, keyphrase extraction is an important step for many downstream tasks, as already described. In this workshop, we aim to cover the foundations of keyphrase extraction in scientific documents and provide a discussion venue for academia and industries on the topic of keyword extraction. Our contributions in the workshop are as follows.

(1.) We make a new use of the existing dataset from the Web of Science (WOS) [34]. This dataset has been used as a benchmark dataset for hierarchical classification systems. Since it comes with reference lists of keywords, we utilize it as a benchmark dataset for keyword extraction. In this workshop, together with the participants, we study the feasibility of that dataset in three systems.

(2.) We introduce three commonly used systems in academia and industry for keyword extraction. For the various use cases of keyword extraction, we also design baseline evaluation metrics for each system.

(3.) We encourage participants to discuss, extend, and evaluate the systems that we have introduced.

System Design of Keyword Extraction. For the keyword extraction, we provide two systems based on the unsupervised, graph-based algorithm TextRank [35]. System 1 (see Section 4) is to develop the TextRank keyword extractor from scratch in order to understand the reasoning behind it. System 2 (see Section 5) combines the TextRank algorithm with the K-Means clustering algorithm [36, 37] to provide keyphrases for each specific field ("cluster"). In System 3 (see Section 6), we cover the NER task, where an entity in the sentence is identified as person, organization, and others from predefined categories. We will focus primarily on the biomedical domain using the state-of-the-art biomedical NER tool called HunFlair [38]. We also provide some baseline NERs for participants to evaluate.

Beyond this workshop, the keyphrase extraction and NER methods we present are applicable to other text corpora, including media texts and legal texts; one only has to aware the domain-specific nature and properly adjust the algorithm pipeline. As such, we have linked the 20 newsgroup text dataset for the participants to try their keyphrase extraction system on.

2. Benchmark Dataset

We take a subset of 46,985 records from the Web of Science dataset (WOS). The original WOS dataset is provided by Kamran Kowsari in the *HDLTex: Hierarchical Deep Learning for Text Classification* paper [34]. The original data was provided in .txt format.

For the ease of work, we have pre-processed the original data and store it into .csv dataframe format, which would be most compatible with our Python working setup. The final dataframe is in the format as in Table 1, where (1) each record corresponds to a single scientific document, and (2) has the following columns:

- `Domain`: the domain the document belongs to,
- `area`: the sub-domain the document belongs to,
- `keywords`: the list of keyphrases provided by the authors, stored as a single string with separator ";",
- `Abstract`: the abstract of the document.

Columns `Y1` and `Y2` which are simply the index of column `Domain` and `area`, respectively. Column `Y` are the

Y1	Y2	Y	Domain	area	keywords	Abstract
5	50	122	Medical	Sports Injuries	Elastic therapeutic tape; Material properties; Tension test	The aim of this study was to analyze stabilometry in athletes...
5	48	120	Medical	Senior Health	Sports injury; Athletes; Postural stability	This study examined the influence of range of motion of the ankle joints on elderly people's balance ability...

Table 1
A sample of the WOS benchmark dataset.

sub-sub-domain, which we do not use here but includes for reference.

In the corpus, we are provided with scientific articles from seven domains: Medical, Computer Science (CS), Biochemistry, Psychology, Civil, Electronics and Communication Engineering (ECE), and Mechanical and Aerospace Engineering (MAE). Therefore, column `Y1` consists of unique values from 0 to 6.

In Table 1, note that both records have the same domain `Y1` as "5" corresponding to `Domain` as "Medical". Their sub-domain `Y2` differs: the first record is about "Sports Injuries", while the second record is about "Senior Health". `keywords` and `Abstract` of each record match its sub-domain.

Finally, the records are splitted at the ratio 70:30 into the train/test sets with 32,899 and 14,096 abstracts, respectively. We provide the training set *with* `keywords` column to the participants for the training of their keyword and/or NER extraction system, and the test set for the participants to evaluate the system. The reason for splitting the dataframe is so that the participants do not overfit their system towards the whole dataset. We encourage them to design their system based on the features learnt from the training set and apply the identical pipeline to the test set.

3. Systems

Now we discuss the three systems we provide to the participants as simple baselines for keyword extraction using the benchmark dataset. Certainly, there are various possible extensions to them. We list the participant contributions under Section 7.

4. System 1: TextRank Algorithm

In System 1, we build the TextRank algorithm from scratch and add customizations to our needs, e.g., filtering by Part-of-Speech tags.

4.1. TextRank

The TextRank algorithm is a graph-based algorithm which, as the name suggests, is used to assign scores to texts, thereby giving a ranking [35]. It has numerous use cases in the NLP domain including webpage ranking (better known as PageRank), extractive text summarization, and keyword extraction [35, 39, 19, 17, 40, 41]. Across different use cases, the base TextRank algorithm remains the same; one only needs to adjust what is designated as nodes, edges, and edge weights when constructing the graph from the text corpus. The higher edge weight means the higher chance of choosing this particular edge to proceed to the next node. For example, in the web context, the PageRank Algorithm considers different webpages as nodes and the hyperlinks between webpage pairs as edges. Here, the edges are asymmetrically directed, since there could be a hyperlink from one page to another but not necessarily vice versa. The edges can then be weighted by the number of hyperlinks.

In our keyword extraction, the TextRank algorithm works by considering terms in text as graph nodes, term co-occurence as edges, and the number of co-occurence of two terms within a certain window as the edge weights. Note that the co-occurence window is a fixed pre-specified size (say, 5-gram within sentence boundary). Based on this notion, the graph is treated as weighted but undirected.

Subsequently, each term score is given by how "likely" an agent, starting at a random point in the graph and continuously jumping along the weighted edges, will end up at that term node after a long time horizon. The terms with higher scores are then considered more important, that is, the "keywords" extracted by the TextRank system.[1]

4.2. Implementation

We implement a very basic keyword extraction system based on the TextRank algorithm from scratch, in order for the participants to get hands-on experience on how the algorithm works. Subsequently, we propose additional improvement ideas so that participants have the opportunity to be creative and improve the basic system.

For implementation, we mainly use the Python package for natural language processing called `spaCy` [42]. spaCy utilizes pre-trained language models to perform many NLP tasks, among other things, Part-of-Speech tag-

[1] In the web analogy, the webpage score would correspond to the chance that an Internet user would end up in that webpage after continuously browsing through the hyperlinks. In this sense, we retrieve the most popular webpages.

ging (PoS tagging), semantic dependency parsing, and Named-Entity Recognition. In our case, we use spaCy along with its small pre-trained model for English language (`en_core_web_sm`) as a text pre-processor and tokenizer. The rest of tasks are handled by usual built-in Python libraries.

Our basic system consists of the following steps:

(1.) Text pre-processing: stopword and punctuation removal.

(2.) Text tokenization: tokenizing the text and build a vocabulary list.

(3.) Build the adjacency matrix from the graph.
- Matrix index in row and column: terms in the vocabulary list.
- Matrix entries: co-occurence of term pairs within the same window of pre-specified size.

(4.) Normalize the matrix and compute the stationary distribution of the matrix.

(5.) Retrieve keyword(s) corresponding to terms with highest stationary probabilities.

The implemented code is stored as a Jupyter notebook and hosted on Google Colaboratory and allows the participant to test and work directly on the code online without local installation. There, the step-by-step description is provided and a code sanity check was performed. For example, our system extracts valid keywords "cute", "dog", "cat" (in descending order by term prominence) for a short text: "This is a very cute dog. This is another cute cat. This dog and this cat are cute".

4.3. Further Ideas

Inspired by existing keyword extraction systems in Python such as `summa` [43] and `pke` [44], we have provided participants with a list of ideas to further improve the keyword extraction system along with hints for Python implementation using spaCy (see the Jupyter notebook):

- Improve the pre-processing step:
  - Remove numbers.
  - Standardize casings, such as lower-casing the entire text.
  - Use a domain-specific or custom-made stopword list.
- Improve the tokenization step:
  - Filter by Part-of-Speech tags to only include nouns in the vocabulary list.
  - Use a domain-specific tokenizer such as ScispaCy [45] for biomedical data.
  - Lemmatize or stem tokens before recording them in the vocabulary list and building the adjacency matrix, so that different versions of the same words (such as plural "solitons" and singular "soliton") are mapped to the same record.
- Add the post-processing step:
  - Exclude keywords that are too short.
- Agglomerate keywords (and perhaps add back some stopwords) to form "keyphrases" ("the" and "of" should not be removed within "the Department of Health").

Advanced participants are also directed to another Python package `NetworkX`, which has a built-in, computationally efficient implementation for the TextRank algorithm [46].

4.4. Evaluation: Instance-Based Performance

In System 1, the objective is *instance-based*, that is, for each abstract, we need to evaluate how well the algorithm performs. The metric could be accuracy, that is, the ability to find as many keyphrases (compared to the reference list) as possible. We can also compute the precision and recall scores (micro or macro). We provide a simple baseline evaluation function in the notebook. Here, we allow fuzzy matching algorithms on the phrase level, where the cut-off ratio and the edit distance between the candidate term and the reference term can be adjusted.

5. System 2: TextRank with Clustering

In System 2, we extend the TextRank keyword extraction described in System 1 (see Section 4) and apply it to a group of texts clustered by the K-Means algorithm. In this way, we obtain a more focused keyword list specifically for each text group and learn about its characteristics.

5.1. K-Means Algorithm

The K-Means algorithm is a clustering algorithm which partitions points in a vector space into "K" clusters ("K" being pre-specified), such that each point belongs to the cluster with the nearest cluster centroid (called "Means") [36, 37]. It works in the following steps.

(1.) Assign k random points as the cluster "means".

(2.) Doing the following until the convergence:

a) Assignment step: Assign each point to the cluster with the least squared Euclidean distance to the cluster mean,

b) Update step: Recalculate the "mean" as the average of all the points assigned to each cluster,

c) Terminate when the cluster assignment stabilizes.

We ultimately choose the K-Means algorithm for clustering because of its low complexity: it works very fast for large datasets like ours [47, 48]. Often, one hidden caveat about the K-Means algorithm is the choice of the number of clusters "K". However, in our specific use case with the scientific publications, we usually have a good estimate based on the number of target disciplines. Therefore, K-Means serves our purpose well.

5.2. Preprocessing: Sentence-BERT Embeddings

As mentioned in the previous section, K-Means clusters points in a vector space. Therefore, we need to transform each text in our dataset into a vector representation. This is often done by averaging pre-trained word embeddings over all the words that appear in the document, regardless of whether they are context-free embeddings like GloVe [49] or contextualized embeddings like BERT [50]. However, this has been shown to perform worse than directly deriving contextualized sentence embeddings (Sentence-BERT [51]). Therefore, we opt for contextualized sentence embeddings from Sentence-BERT, which is trained on the Siamese BERT networks [51]. More technical details can be found in the original paper by N. Reimers and I. Gurevych [51].

The Sentence-BERT transforms each text into a 384-dimensional semantically meaningful vector, which is now ready to be an input to the K-Means algorithm for clustering.

5.3. Implementation

We add the clustering step to our pipeline, which effectively results in the following procedure:

(1.) For each document, extract its Sentence-BERT embedding,

(2.) Cluster the documents into K groups based on their Sentence-BERT embeddings, i.e., by the sentence contents,

(3.) For each document cluster, extract its keyphrases.

First, we generate embedding representations for each text, which is very easy by the Python package `sentence-transformers`. The package `sentence-transformers` offers several pre-trained models for different purposes, from which we choose the small model (`all-MiniLM-L6-v2`).

Second, to group the documents, we use the implementation in the package `sklearn` [52]. Furthermore, we provide a cluster visualization using the package `matplotlib` [53]. We set the parameter $K = 7$ for the K-Means algorithm, which is the number of disciplines in the WOS dataset.

Finally, we extract the keyphrases from each cluster. Unlike in System 1, we do not implement the TextRank algorithm from scratch, but instead use the existing Python package `pke` [44]. `pke` provides implementations of numerous keyword extraction algorithms from publications, as well as allowing customization such as Part-of-Speech tag filters and the limit on the maximum number of words in a single keyphrase. In our case, we simply use the basic TextRank algorithm, also to demonstrate that even the very basic algorithm can already yield satisfying outputs.

Like in System 1, the code implemented for System 2 is stored as a Jupyter notebook and hosted on Google Colaboratory. The step-by-step description is provided, and a code sanity check succeeds at characterizing a cluster: the cluster mostly consisting of medical articles has relevant keyphrases such as "patient group", "treatment effects", "autism patient" among the top-10 extracted keyphrase list.

5.4. Further Ideas

We invite participants to explore improvement ideas and provide coding hints on how to implement them on `pke`:

- Customize the TextRank algorithm:
  - Change the window size.
- Use alternative keyword extraction algorithms to the TextRank algorithm, such as:
  - The TopicRank algorithm [54],
  - The Multipartite algorithm [55],
  - The BERTopic algorithm [56].
- Impose extra criteria on valid keyphrases, such as:
  - Change the maximum number of words allowed in a single keyphrase,
  - Restrict the keyphrase to only contain the top certain percentage of all keywords.

5.5. Evaluation: Cluster-Based Performance

Using a similar evaluation function as in System 1 (See Section 4.4), we now look at a *cluster-based* objective.

This means that we take all the keywords from the articles clustered in the same group and build a new reference list of keywords. Subsequently, the evaluation of the user-generated list will be compared with this expanded list. Notably, this approach increases the coverage of keywords in the reference, in the hope of covering more out-of-abstract keywords in this expanded list. However, it comes at the cost of increasing the denominator when we compare the user-generated list to the reference list. One way to better present the reference list of one cluster is to process the list by criteria such as frequency. Another way to evaluate is using word embedding similarities (c.f. KeyBert [33] as an example of leveraging embeddings). In this way, we have a better view of the extracted keywords and the degree to which the user-generated list is close to the reference list. In particular, this technique is useful for assessing the difference set between the user-generated list and the reference one.

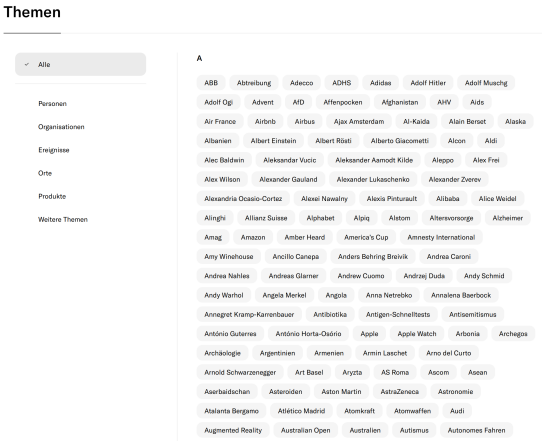


Figure 2: NZZ Topic Page based on keywords and named entities from news articles. Accessible at nzz.ch/themen.

6. System 3: Named-Entity Recognition as Keyword Extraction

The goal of system 3 is to emulate some of the constraints that may exist in a practical setting. These could be situations where a keyword extractor system cannot be implemented as the output of these systems may be incorrect or non-sensical. Another situation could be that one is required to use existing tools such as a Named-Entity Recognition system and must enact measures to improve the output of the model.

6.1. Named Entities, Named-Entity Recognition and Keyword Extraction

A named entity (NE) in most cases is a proper noun, the most common categories being person, location and organization; however, other categories that are not proper nouns, such as temporal expressions, are also possible. Named-Entity Recognition consists of locating and classifying named entities mentioned in unstructured text into predefined categories [57, Chapter. 8.3]. Keywords are single or multi-word expressions that under ideal circumstances should concisely represent the key content of a document [58, Page 3]. As the goal of NER is to assign a label to spans of text [57, Chapter. 8.3], it is a classification task that can be solved by building a machine learning model [59].

The difference between keyword extraction and NER is as follows. Named entities are words or phrases with a specific label determined by predefined classes of a given NER model. Therefore, these entities may not necessarily represent the essential content of a document. Keywords are not limited by the fixed categories of an NER model, and may contain named entities if those entities are representative of a given document. For example, a document about Heathrow Airport can contain keywords such as "arrival", "customs", "departure", "duty free", "immigration" and "London". Depending on the model classes, an NER model on the same text could extract entities such as "British Airways" (ORG), "London" (LOC), "United Kingdom" (LOC), etc. In this example, there is overlap between the keywords and named entities; however, due to the defining characteristics of both approaches, there is a significant difference between the lists.

Figure 2 demonstrates the use of keyword extraction and named-entity recognition in the industry setting at Neue Zürcher Zeitung (NZZ), where key terms are extracted and relevant articles are assigned to the terms.

6.2. Use of Keywords in the News Domain

As mentioned above, for a given text, keywords and the output of a NER model may overlap. When it comes to analyzing news, a typical NER model (with common categories such as person, organization, and location) excels at finding named entities for the model-specific categories. However, only extracting the entities is inadequate for finding nuanced differences between multiple articles that contain identical named entities. In Table 2 we see the titles of 10 articles published in Neue Zürcher Zeitung (NZZ) during March 2022. According to the NER model for German texts used internally by the NZZ, all articles have "Ukraine" (location) as a common named entity. Despite the similarities, there are thematic differences between these articles. After using a keyword extraction system that uses similar methodologies mentioned in Systems 1 and 2, keywords that are not named

Number	NZZ Article Title
1	Eine Zürcherin nimmt ukrainische Flüchtlinge auf – und fühlt sich vom Staat alleingelassen «Eine Solidaritätsbekundung auf Instagram zu posten, reicht nicht»:
2	Viele Zürcherinnen und Zürcher möchten Flüchtlinge aus der Ukraine bei sich zu Hause aufnehmen
3	150 Ukraine-Flüchtlinge sind im Kinderdorf – wie geht es weiter?
4	Krieg in der Ukraine: Wie ein SVP-Dorf Flüchtlinge aufnimmt
5	Neutralität im Ukraine-Krieg - wo genau steht die Schweiz?
6	Neutralität: Fand in der Schweiz gerade eine Zeitenwende statt
7	Putin, die Schweiz und die zwei Seiten der Neutralität
8	Christoph Blocher: Neutralität ist nicht nur Selbstzweck
9	Sicherheitspolitik: Militärische Neutralität weiterdenken
10	Sicherheitspolitik: Solidarische Neutralität

Table 2
Titles of 10 articles published in Neue Zürcher Zeitung (NZZ) during March 2022.

entities were found. These keywords demonstrate thematic groupings between the articles. The most common keyword for articles 1-4 is "Flüchtlinge" ("refugees"), and for articles 5-10 is "Neutralität" ("neutrality"). This difference can also be observed in the article titles, and upon closer inspection of the article content, it is evident that some of the articles (1-4) revolve around the topic of refugees from Ukraine, while other articles (5-10) discuss the notion of neutrality. Using named entities or, in some cases, a predefined list of keywords can be useful to define broad topic pages (see nzz.ch/themen), but keywords offer concise yet semantically insights into the content of a document. Therefore, they can be potentially used to automatically identify possible subtopics with a news story or discover emerging topics from newly published articles.

6.3. Data Preparation

The FLAIR framework [60] was chosen as it contains many out-of-the-box NER models for generic and biomedical texts. Furthermore, the framework is also useful for integrating pre-trained embeddings and models. As many of the texts are from the biomedical domain, the ScispaCy library was used for word and sentence tokenization [61]. The results of the NER models were given to the participants. The *ner-english* model is a 4-class NER model for English, which comes with FLAIR [62]. This model has the following categories: locations (LOC), persons (PER), organizations (ORG), and miscellaneous (MISC) [63]. We also provided participants with NER results from HunFlair [38], which is an NER tagger for biomedical texts. This biomedical NER tagger is based on the HUNER tagger, and has the follwing named-entity categories: Chemicals, Diseases, Species, Genes or Proteins, and Cell lines [64]. As an additional hint to participants, document embeddings for each item in the train and test sets, as well as word embeddings for the entire corpus, were generated from a fastText model[2] trained on the English Common Crawl dataset (`cc.en.300.bin`)[3].

[2]https://fasttext.cc/ (last accessed: June 20, 2022).
[3]https://fasttext.cc/docs/en/crawl-vectors.html (last accessed: June 20, 2022).

6.4. Pre-Trained NER Models

There are some disadvantages to using pre-trained NER models. One should take into consideration that using a pre-trained model to extract named entities out of documents from different domains can result in a fall in model performance [65]. The training data and categories of the model will influence the output. For example, the string "ATP" can be labeled as an organization (e.g. Association of Tennis Professionals) by one model and as a chemical (e.g. adenosine triphosphate) by a biomedical-NER model. Creating an NER model for a specific type of entity requires the annotation of a corpus, which can be a significant expense and effort for the user [65].

6.5. Further Ideas

The challenge of this system lies in working with pre-calculated data from systems that cannot be influenced. The participants are provided with multiple tables with the output of two different NER systems, fastText document, and word vectors (see Section 6.3). In addition, they also have a table at their disposal to verify whether a keyword for a given document is present in the abstract and whether it was discovered by any of the NER models (with 100% string matches). The intuition of System 3 is that given the resources (cost, time, hardware), one needs to come up with the best possible strategies to detect meaningful keywords.

6.6. Evaluation: Instance-based Performance

In addition to the pre-calculated data, the participants were also given evaluation functions to compare differences between their system NER model output and the keyword list that came with the documents. There are cases where an item from the curated keyword list does not contain the keyword in the abstract, or contains a partial or inflected form of the keyword. The evaluation function contains a partial string matching sequence, where one can choose the amount of character similarity between two strings. For example, a document has the label "radio frequency", but the string "radio frequencies" is present in the abstract and the inflected form was also found by one of the NER models. For this case, participants can set a string similarity value (e.g., 80% similarity) to circumvent the issues caused by inflected forms, or partially mentioned forms ("radio frequency" vs. "radio frequency scanner"). Using the resources at their disposal, participants must develop the best possible strategies to build a system that can detect the maximum number of relevant keywords.

7. Participant Contributions

Our participants have further investigated keyphrase extractions in System 1 and provided valuable contributions to our proceedings. Their original theses can be founded at the following Google Drive folder.

The basic TextRank keyword extractor in System 1 has been extended to account for the following data preprocessing steps: (1) remove numbers; (2) restrict valid keywords to only nouns; (3) restrict valid keywords by imposing the minimum string length. The contribution can be found on the Google Drive folder.

Additionally, the evaluation system has been generalized to output numerical performance scores, allowing simpler comparisons of different keyword extractors. The contribution can be found on the Google Drive folder.

Finally, a comparison between the TextRank algorithm and further unsupervised keyphrase extraction methods has been provided. The limitation of TextRank is that it only considers the co-occurences of the word pair and not the semantical meanings, which may cause certain extracted "frequent" word pairs to either be irrelevant or under-represented. Therefore, an experiment has been performed using the `pke` library to compare the performance of the TextRank algorithm and several other unsupervised keyphrase extraction algorithms on the benchmark test dataset. The contribution can be found on the Google Drive folder.

Beyond the academic setting, the use of keyword extractions is demonstrated in the industry setting, where Wyona AG utilizes keyword extractors in the working pipeline of the Q&A Chatbot "Katie". The contribution can be found on the Google Drive folder.

8. Conclusion

In this workshop, we provided the background and baseline systems for keyword extraction, shared a benchmark dataset on scientific keyword extraction, and invited contributions from participants from industry and academia. The methodologies discussed can be extended to keyword extraction in other domains (e.g., legal and news).

Acknowledgements

The authors would like to thank the organizers from SwissText2022 for hosting our workshop. Peter Egger and the Chair of Applied Economics acknowledge the support of the Department of Management, Technology, and Economics at ETH Zurich. Ce Zhang and the DS3Lab gratefully acknowledge the support from the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number MB22.00036 (for European Research Council (ERC) Starting Grant TRIDENT 101042665), the Swiss National Science Foundation (Project Number 200021_184628, and 197485), Innosuisse/SNF BRIDGE Discovery (Project Number 40B2-0_187132), European Union Horizon 2020 Research and Innovation Programme (DAPHNE, 957407), Botnar Research Centre for Child Health, Swiss Data Science Center, Alibaba, Cisco, eBay, Google Focused Research Awards, Kuaishou Inc., Oracle Labs, Zurich Insurance, and the Department of Computer Science at ETH Zurich. We would like to thank Neue Zürcher Zeitung for collaborating on this project.

References

[1] T. D. Nguyen, M.-Y. Kan, Keyphrase extraction in scientific publications, in: D. H.-L. Goh, T. H. Cao, I. T. Sølvberg, E. Rasmussen (Eds.), Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 317–326.

[2] S. N. Kim, M.-Y. Kan, Re-examining automatic keyphrase extraction approaches in scientific articles, in: Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications (MWE 2009), Association for Computational Linguistics, Singapore, 2009, pp. 9–16.

[3] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, C. G. Nevill-Manning, Domain-specific keyphrase extraction, in: T. Dean (Ed.), Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages, Morgan Kaufmann, 1999, pp. 668–673.

[4] C. Gutwin, G. Paynter, I. Witten, C. Nevill-Manning, E. Frank, Improving browsing in digital libraries with keyphrase indexes, Decision Support Systems 27 (1999) 81–104. doi:https://doi.org/10.1016/S0167-9236(99)00038-X.

[5] O. Medelyan, I. H. Witten, Domain-independent automatic keyphrase indexing with small training sets, J. Am. Soc. Inf. Sci. Technol. 59 (2008) 1026–1040.

[6] O. Borisov, M. Aliannejadi, F. Crestani, Keyword extraction for improved document retrieval in conversational search, CoRR abs/2109.05979 (2021).

[7] J. Han, T. Kim, J. Choi, Web document clustering by using automatic keyphrase extraction, in: 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops, 2007, pp. 56–59. doi:10.1109/WI-IATW.2007.46.

[8] K. M. Hammouda, D. N. Matute, M. S. Kamel, Corephrase: Keyphrase extraction for document clustering, in: P. Perner, A. Imiya (Eds.), Machine Learning and Data Mining in Pattern Recogni-

tion, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 265–274.

[9] M. Litvak, M. Last, Graph-based keyword extraction for single-document summarization, in: Proceedings of the Workshop on Multi-Source Multilingual Information Extraction and Summarization, MMIES '08, Association for Computational Linguistics, USA, 2008, p. 17–24.

[10] K. Sarkar, A keyphrase-based approach to text summarization for english and bengali documents, Int. J. Technol. Diffus. 5 (2014) 28–38. doi:10.4018/ijtd.2014040103.

[11] J. R. Thomas, S. K. Bharti, K. S. Babu, Automatic keyword extraction for text summarization in e-newspapers, in: Proceedings of the International Conference on Informatics and Analytics, ICIA-16, Association for Computing Machinery, New York, NY, USA, 2016. doi:10.1145/2980258.2980442.

[12] S. N. Kim, O. Medelyan, M.-Y. Kan, T. Baldwin, SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles, in: Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, Uppsala, Sweden, 2010, pp. 21–26.

[13] J. Li, Y. Li, Z. Xue, Keywords extraction algorithm of financial review based on dirichlet multinomial model, in: Y. Jia, W. Zhang, Y. Fu (Eds.), Proceedings of 2020 Chinese Intelligent Systems Conference, Springer Singapore, Singapore, 2021, pp. 107–116.

[14] M. Pejić Bach, Ž. Krstić, S. Seljan, L. Turulja, Text mining for big data analysis in financial sector: A literature review, Sustainability 11 (2019). doi:10.3390/su11051277.

[15] M. Jungiewicz, M. Łopuszyński, Unsupervised keyword extraction from polish legal texts, in: A. Przepiórkowski, M. Ogrodniczuk (Eds.), Advances in Natural Language Processing, Springer International Publishing, Cham, 2014, pp. 65–70.

[16] D. Wu, W. Uddin Ahmad, S. Dev, K.-W. Chang, Representation Learning for Resource-Constrained Keyphrase Generation, arXiv e-prints (2022). arXiv:2203.08118.

[17] J. Piskorski, N. Stefanovitch, G. Jacquet, A. Podavini, Exploring linguistically-lightweight keyword extraction techniques for indexing news articles in a multilingual set-up, in: Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation, Association for Computational Linguistics, Online, 2021, pp. 35–44.

[18] S. Suzuki, H. Takatsuka, Extraction of keywords of novelties from patent claims, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, The COLING 2016 Organizing Committee, Osaka, Japan, 2016, pp. 1192–1200.

[19] J. Hu, S. Li, Y. Yao, L. Yu, G. Yang, J. Hu, Patent keyword extraction algorithm based on distributed representation for patent classification, Entropy 20 (2018). doi:10.3390/e20020104.

[20] H. Ding, X. Luo, Attention-based unsupervised keyphrase extraction and phrase graph for covid-19 medical literature retrieval, ACM Trans. Comput. Healthcare 3 (2021). doi:10.1145/3473939.

[21] M. Komenda, M. Karolyi, A. Pokorná, M. Víta, V. Kríž, Automatic keyword extraction from medical and healthcare curriculum, in: 2016 Federated Conference on Computer Science and Information Systems (FedCSIS), 2016, pp. 287–290.

[22] Q. Li, Y.-F. B. Wu, Identifying important concepts from medical documents, Journal of Biomedical Informatics 39 (2006) 668–679. doi:https://doi.org/10.1016/j.jbi.2006.02.001.

[23] A. Zehtab-Salmasi, M.-R. Feizi-Derakhshi, M.-A. Balafar, Frake: Fusional real-time automatic keyword extraction, 2021. arXiv:2104.04830.

[24] C. Sun, L. Hu, S. Li, T. Li, H. Li, L. Chi, A review of unsupervised keyphrase extraction methods using within-collection resources, Symmetry 12 (2020). doi:10.3390/sym12111864.

[25] D. Mahata, R. R. Shah, J. Kuriakose, R. Zimmermann, J. R. Talburt, Theme-weighted ranking of keywords from text documents using phrase embeddings, in: 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2018, pp. 184–189. doi:10.1109/MIPR.2018.00041.

[26] S.-C. Kuai, W.-H. Liao, C.-Y. Chang, G.-J. Yu, Fbkea: A feature-based keyword extraction algorithm for improving hit performance, in: 2021 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), 2021, pp. 1–2. doi:10.1109/ICCE-TW52618.2021.9602870.

[27] R. Saga, H. Kobayashi, T. Miyamoto, H. Tsuji, Measurement evaluation of keyword extraction based on topic coverage, in: C. Stephanidis (Ed.), HCI International 2014 - Posters' Extended Abstracts, Springer International Publishing, Cham, 2014, pp. 224–227.

[28] F. Liu, X. Huang, W. Huang, S. X. Duan, Performance evaluation of keyword extraction methods and visualization for student online comments, Symmetry 12 (2020). doi:10.3390/sym12111923.

[29] L. Bornmann, R. Mutz, Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references, Journal of the Association for Information Science and Technology : JASIST 66 (2015-11) 2215 – 2222. doi:10.1002/asi.23329, published online 29 April 2015.

[30] B. Hua, Y. Shin, Extraction of sentences describing originality from conclusion in academic papers, in:

Y. Zhang, C. Zhang, P. Mayr, A. Suominen (Eds.), Proceedings of the 1st Workshop on AI + Informetrics (AII2021) co-located with the iConference 2021, Virtual Event, March 17th, 2021, volume 2871 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021, pp. 58–70.

[31] M. Krallinger, F. Leitner, O. Rabal, M. Vazquez, J. Oyarzábal, A. Valencia, Chemdner: The drugs and chemical names extraction challenge, Journal of Cheminformatics 7 (2015) S1 – S1.

[32] M. F. Porter, An Algorithm for Suffix Stripping, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997, p. 313–316.

[33] M. Grootendorst, Keybert: Minimal keyword extraction with bert., 2020. doi:10.5281/zenodo.4461265.

[34] K. Kowsari, D. E. Brown, M. Heidarysafa, K. Jafari Meimandi, , M. S. Gerber, L. E. Barnes, Hdltex: Hierarchical deep learning for text classification, in: Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on, IEEE, 2017.

[35] R. Mihalcea, P. Tarau, TextRank: Bringing order into text, in: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 404–411.

[36] S. Lloyd, Least squares quantization in pcm, IEEE Transactions on Information Theory 28 (1982) 129–137. doi:10.1109/TIT.1982.1056489.

[37] J. MacQueen, Classification and analysis of multivariate observations, in: 5th Berkeley Symp. Math. Statist. Probability, 1967, pp. 281–297.

[38] L. Weber, M. Sänger, J. Münchmeyer, M. Habibi, U. Leser, A. Akbik, HunFlair: an easy-to-use tool for state-of-the-art biomedical named entity recognition, Bioinformatics 37 (2021) 2792–2794. doi:10.1093/bioinformatics/btab042.

[39] J. Son, Y. Shin, Music lyrics summarization method using textrank algorithm, Journal of Korea Multimedia Society 21 (2018) 45–50. doi:https://doi.org/10.9717/kmms.2018.21.1.045.

[40] C. Wu, L. Liao, F. Afedzie Kwofie, F. Zou, Y. Wang, M. Zhang, Textrank keyword extraction method based on multi-feature fusion, in: X.-S. Yang, S. Sherratt, N. Dey, A. Joshi (Eds.), Proceedings of Sixth International Congress on Information and Communication Technology, Springer Singapore, Singapore, 2022, pp. 493–501.

[41] S. Pan, Z. Li, J. Dai, An improved textrank keywords extraction algorithm, in: Proceedings of the ACM Turing Celebration Conference - China, ACM TURC '19, Association for Computing Machinery, New York, NY, USA, 2019. doi:10.1145/3321408.3326659.

[42] I. Montani, M. Honnibal, M. Honnibal, S. V. Landeghem, A. Boyd, H. Peters, P. O. McCann, M. Samsonov, J. Geovedi, J. O'Regan, D. Altinok, G. Orosz, S. L. Kristiansen, D. de Kok, L. Miranda, Roman, E. Bot, L. Fiedler, G. Howard, Edward, W. Phatthiyaphaibun, R. Hudson, Y. Tamura, S. Bozek, murat, R. Daniels, P. Baumgartner, M. Amery, B. Böing, explosion/spaCy: New Span Ruler component, JSON (de)serialization of Doc, span analyzer and more, 2022. doi:10.5281/zenodo.6621076.

[43] F. Barrios, F. López, L. Argerich, R. Wachenchauzer, Variations of the similarity function of textrank for automated summarization, CoRR abs/1602.03606 (2016). arXiv:1602.03606.

[44] F. Boudin, pke: an open source python-based keyphrase extraction toolkit, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations, Osaka, Japan, 2016, pp. 69–73.

[45] M. Neumann, D. King, I. Beltagy, W. Ammar, ScispaCy: Fast and robust models for biomedical natural language processing, in: Proceedings of the 18th BioNLP Workshop and Shared Task, Association for Computational Linguistics, Florence, Italy, 2019, pp. 319–327. doi:10.18653/v1/W19-5034.

[46] A. Hagberg, P. Swart, D. S Chult, Exploring network structure, dynamics, and function using networkx (2008).

[47] D. Xu, Y. Tian, A comprehensive survey of clustering algorithms, Annals of Data Science 2 (2015) 165–193. doi:10.1007/s40745-015-0040-1.

[48] A. E. Ezugwu, A. M. Ikotun, O. O. Oyelade, L. Abualigah, J. O. Agushaka, C. I. Eke, A. A. Akinyelu, A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects, Engineering Applications of Artificial Intelligence 110 (2022) 104743. doi:https://doi.org/10.1016/j.engappai.2022.104743.

[49] J. Pennington, R. Socher, C. Manning, GloVe: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1532–1543. doi:10.3115/v1/D14-1162.

[50] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.

[51] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using Siamese BERT-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3982–3992. doi:10.18653/v1/D19-1410.

[52] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[53] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing in Science & Engineering 9 (2007) 90–95. doi:10.1109/MCSE.2007.55.

[54] A. Bougouin, F. Boudin, B. Daille, TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction, in: International Joint Conference on Natural Language Processing (IJCNLP), Nagoya, Japan, 2013, pp. 543–551. URL: https://hal.archives-ouvertes.fr/hal-00917969.

[55] F. Boudin, Unsupervised keyphrase extraction with multipartite graphs, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 667–672. doi:10.18653/v1/N18-2105.

[56] M. Grootendorst, Bertopic: Neural topic modeling with a class-based tf-idf procedure, arXiv preprint arXiv:2203.05794 (2022).

[57] D. Jurafsky, J. H. Martin, Speech and language processing (draft), preparation [cited 2020 June 1] Available from: https://web. stanford. edu/~ jurafsky/slp3 (2018).

[58] M. W. Berry, J. Kogan, Text mining: applications and theory, John Wiley & Sons, 2010.

[59] A. Mansouri, L. S. Affendey, A. Mamat, Named entity recognition approaches, International Journal of Computer Science and Network Security 8 (2008) 339–344.

[60] A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, R. Vollgraf, FLAIR: An easy-to-use framework for state-of-the-art NLP, in: NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), 2019, pp. 54–59.

[61] M. Neumann, D. King, I. Beltagy, W. Ammar, Scispacy: Fast and robust models for biomedical natural language processing, CoRR abs/1902.07669 (2019).

[62] A. Akbik, D. Blythe, R. Vollgraf, Contextual string embeddings for sequence labeling, in: COLING 2018, 27th International Conference on Computational Linguistics, 2018, pp. 1638–1649.

[63] E. F. Tjong Kim Sang, F. De Meulder, Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition, in: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, 2003, pp. 142–147.

[64] L. Weber, J. Münchmeyer, T. Rocktäschel, M. Habibi, U. Leser, HUNER: improving biomedical NER with pretraining, Bioinformatics 36 (2019) 295–302. doi:10.1093/bioinformatics/btz528.

[65] M. Marrero, J. Urbano, S. Sánchez-Cuadrado, J. Morato, J. M. Gómez-Berbís, Named entity recognition: Fallacies, challenges and opportunities, Computer Standards & Interfaces 35 (2013) 482–489. doi:https://doi.org/10.1016/j.csi.2012.09.004.

A. List of participants to the workshop

We thank our workshop participants for valuable feedback, contributions, and suggestions.

Susie Xi Rao, ETH Zurich (Organizer)
Piriyakorn Piriyatamwong, ETH Zurich (Organizer)
Parijat Ghoshal, NZZ AG (Organizer)
Vanya Brucker, Wyona AG
Andrea Bussolan, SUPSI
Mercedes García Martínez, Pangeanic
Sandra Mitrović, IDSIA USI-SUPSI
Sara Nasirian, SUPSI
Emmanuel de Salis, HE-Arc
Natasa Sarafijanovic-Djukic, FFHS
Dietrich Trautmann, Thomson Reuters
Michael Wechner, Wyona AG
Peter Egger, ETH Zurich (Principal Investigator)
Ce Zhang, ETH Zurich (Principal Investigator)