IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 21, 2021, accepted December 21, 2021, date of publication January 21, 2022, date of current version January 31, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3144932

NeSyChair: Automatic Conference Scheduling Combining Neuro-Symbolic Representations and Constrained Clustering

TADEJ ŠKVORC[1,2], NADA LAVRAČ[2], AND MARKO ROBNIK-ŠIKONJA[1]
[1]Faculty of Computer and Information Science, University of Ljubljana, 1000 Ljubljana, Slovenia
[2]Jožef Stefan Institute, 1000 Ljubljana, Slovenia

Corresponding author: Tadej Škvorc (tadej.skvorc@fri.uni-lj.si)

This work was supported in part by the Slovenian Research Agency (ARRS) through a Young Researcher Grant and Research Core under Grant P6-0411 and Grant P2-0103, and in part by the European Union's Horizon 2020 Research and Innovation Program through the Project Cross-Lingual Embeddings for Less-Represented Languages in European News Media (EMBEDDIA) under Agreement 825153.

ABSTRACT Creating the schedule for an academic conference is a time-consuming task. A typical conference schedule consists of sessions containing papers addressing the same research topic. To construct a schedule, conference papers must be grouped according to their research topic, and the obtained groups should fit the assigned time slots. This paper proposes an approach to automating the schedule-creation process. We use multilingual, neuro-symbolic paper representations and novel constrained clustering to group papers into clusters of predetermined size with the same topic fitting the schedule structure. In the process, we combine machine-learning, natural language processing, network analysis, and combinatorial optimization. We tested the components of the proposed approach on a newly created database of papers from six machine learning conferences, which were manually labeled by their research topics. The entire system was tested on two real-world conferences in a multilingual setting. The developed methodology is incorporated into an interactive automatic conference-scheduling system NeSyChair (Neuro-Symbolic Conference Chair), which can be used to create and improve conference schedules.

INDEX TERMS Machine learning, natural language processing, optimization, supervised learning, unsupervised learning.

I. INTRODUCTION

Organizing a scientific conference is a time-consuming task, and so any automation would be welcomed by the conference organizers. While there are several systems that assist organizers in managing the review process, the scheduling of paper presentations remains largely a manual task. A typical conference schedule consists of multiple plenary and parallel sessions, consisting of paper presentations covering the same or similar research topics. To construct a conference schedule, papers are usually first grouped according to their topics and then assigned to available time slots. Doing this manually is time-consuming, as large conferences have presentations of hundreds of papers that address many research topics, while authors, organizers, and venues can impose additional constraints.

The associate editor coordinating the review of this manuscript and approving it for publication was Arianna Dulizia.

Along with the papers consisting of text and metadata, e.g., authors and keywords, conference organizers might have access to additional metadata, often available in the form of networks. Examples of such data are citation networks, where two papers are connected if one of them cites the other, or co-bidding networks, where two papers are connected if the same reviewer bids to review them. This additional information can be useful for finding papers with similar research topics, provided that the information is first converted into a vector form that is suitable for automatic data processing by machine-learning (ML) algorithms, such as clustering.

We propose an automatic approach for the assignment of papers into conference slots using ML, natural language processing (NLP), network analysis, and combinatorial optimization. Our goal is to construct a schedule where the papers from the same research topic are not presented at the same time, enabling participants to attend the presentations of all the papers from their research field. To achieve this goal,

we test several semantic-similarity techniques to identify the papers with similar research topics and propose a novel constrained clustering algorithm that assigns papers to a predefined schedule structure so that each time slot contains similar papers.

While several conferences require the authors of papers to specify the research field during the submission, these fields are often imprecise and do not accurately reflect all the topics, especially in large research fields with many areas of research (e.g., deep learning). Therefore, an automatic neuro-symbolic approach, combining numerical features obtained by the neural embedding of paper texts and symbolic features obtained from paper meta descriptors and the accompanying metadata might be better at grouping papers into relevant topics. Our approach first extracts the relevant information from the papers and the accompanying metadata, present in the form of a graph. We use neural embeddings to transform a paper's text into a numerical vector format and extract additional symbolic features describing the useful properties of the paper's content. Network-based metadata are also represented as numerical vectors using network-analysis methods. Text- and network-based data are then fused into a joint representation, which is used to find similar papers with the proposed constrained-clustering algorithm.

To detect similar papers, we tested several modern neural word-, sentence-, and document-embedding approaches. Embeddings transform the text into numerical vectors so that the vectors contain their semantic information. Recent word-embedding approaches, e.g., ELMo [1] and BERT [2] are known to perform well in a variety of NLP tasks. As these embeddings are computed on large unlabeled text corpora, they are suitable for our task, where large labeled datasets do not exist. In our experiments we use three embedding approaches, i.e., doc2vec [3], BERT [4], and Universal Sentence Encoder [5], to obtain embeddings for either entire papers or n-character-long sections. The proposed representation is cross-lingual and can work in either monolingual or multilingual conference scenarios. We show that the proposed embeddings-based approach is suitable for clustering and classifying papers according to their topics.

The embedded papers are assigned to conference slots using a novel variant of the k-means clustering algorithm that uses combinatorial optimization to ensure the resulting clusters fit the schedule structure and minimize the overlap of paper topics. We evaluate several variations of the clustering algorithm on a synthetic dataset and show that it is capable of generating meaningful clusters with various constraints in the conference schedule. We evaluate the final NeSyChair (Neuro-Symbolic Conference Chair) system approach on a novel, manually labeled dataset consisting of papers from several machine-learning conferences and another multilingual conference from the area of natural language processing. To the best of our knowledge, no system comparable to NeSyChair currently exists.

The contributions of this work are as follows.

1) We developed a unified, multilingual, end-to-end approach to automated conference scheduling using a combination of ML, NLP, and network analysis.
2) We developed an approach that uses combinatorial optimization methods to generate clusters of similar papers, followed by a specialized optimization algorithm that applies size constraints on produced clusters to ensure that the returned sessions match the structure of the conference.
3) We made available a manually topic-labeled, English dataset containing papers from several ML conferences, data from a large, real-world ML conference, and a similar smaller dataset in Slovene (a low-resource language).
4) We evaluated our system on a synthetic dataset as well as on two real-world scenarios and show that our approach works for ML conferences in languages other than English.

The paper is structured into four subsequent sections. The related work is presented in Section II, with the proposed methodology for automatic conference scheduling in Section III. In Section IV, we present the evaluation setting, the analysis of the proposed components, and the evaluation of the entire system. Section V draws the conclusions and presents the ideas for future work.

II. RELATED WORK

There are several conference-management systems that assist conference organizers in organizing and managing conferences. These include EasyChair [6], OpenConf [7], Microsoft Conference Management Toolkit [8], IAPR Conference Management System [9], and EDAS: Editor's assistant [10]. Such systems assist organizers in the paper submission-and-reviewing process, but, in contrast to our system, they do not facilitate automatic conference-schedule construction.

In this section, we briefly review the three main components of the proposed conference-scheduling approach: paper similarity measures, graph-based features for paper similarity, and constrained-clustering methods that can be used to assign papers to a conference schedule.

A. PAPER SIMILARITY

In the area of information retrieval, text similarity is extensively researched [11]. A classic approach is to use a sparse text representation approach, such as bag-of-words [12], to represent a given collection of documents and apply a similarity measure, e.g., cosine similarity, to compute the pairwise document similarity. Hurtado *et al.* [13] use language models generated on abstracts to find similar papers. Besides the full text of abstracts and papers, some authors use additional metadata like keywords to improve the results [14].

Another approach, well-suited to determine the similarity of research papers, is based on terminology extraction. Research papers contain a large amount of scientific terminology specific to their field. Since the terminology is

closely linked to the papers' topics, focusing on the terminology instead of words from the general vocabulary can be useful. Milios *et al.* [15] show that using automatically extracted terminology works well for finding similar papers. Jiang *et al.* [16] describe an approach for terminology extraction from research papers using keywords and title words as a basis for the terminology and extend their approach by finding similar words. In the work described in this paper, we use the extracted terminology in combination with other features and improve the extraction process with rule-based filters that remove terminology that is either too specific or too general to identify the papers' topics.

Recently, approaches based on dense word embeddings have prevailed. Embeddings such as continuous bag-of-words (CBOW) and skip-gram models implemented in the word2vec tool [17] are capable of extracting semantic information from words and mapping words with similar meaning to similar vectors, which can help in determining text similarity. Recently, contextual word embeddings were shown to improve performance in a variety of natural language processing tasks. Such models include BERT [2] and ELMo [1] embeddings, as well as embeddings produced using the transformer neural architecture [18]. Such models achieved state-of-the-art results on a variety of NLP tasks. However, most of these approaches require large training datasets to achieve a competitive performance. Despite this, Turc *et al.* [4] show that a model that is pre-trained on a large unlabeled dataset can achieve good results on a variety of tasks, even when using a small amount of task-specific, labeled data. Nevertheless, such approaches are designed to produce embeddings for smaller units of text, such as words or sentences, which makes them unfit to compute the similarity between entire documents.

Several authors explored ways of obtaining embeddings for larger sequences of text, such as sentences or entire documents. Sitikhu *et al.* [19] present a variety of approaches for embedding larger units of text. Such embeddings can be directly used for detecting similar papers. In our work, we show that both word embeddings (BERT, word2vec) and document embeddings (doc2vec [3], universal sentence encoder [5]) can be used to find similar papers.

An additional benefit of text embeddings is their ability to handle text in multiple languages using cross-lingual text representations. Lample *et al.* [20] show that it is possible to align text embeddings from multiple languages into the same vector space so that words with similar meanings from multiple languages are grouped together. This allows downstream approaches to handle multiple languages without the use of an explicit translation. We exploit this property to build and test a system that can construct schedules in multiple languages.

B. NETWORK ANALYSIS FOR PAPER SIMILARITY

Frequently, pairwise similarities of documents, computed, e.g., by the cosine similarity measure, are used to construct a network of similar papers, followed by their analysis using network analytic techniques. In citation networks, used by Price *et al.* [21], the nodes are individual papers, and each paper is connected to every paper it cites. In networks of bibliographic coupling [22], two papers are connected if they cite a common paper. In networks of co-citations [23], two papers are connected if they are both cited by the same paper. Giles *et al.* [24] present an algorithm called CCIDF (Common Citation $\times$ Inverse Document Frequency), which improves citation networks by weighting citations with the inverse frequency of citations in the entire database. In addition to networks constructed from citations, other networks can be used to find similar papers. Hamasaki *et al.* [25] show that interpersonal networks can be useful for this task.

For analyzing conference papers in the context of constructing a conference schedule, citation-network approaches cannot be used, as the papers that are to be presented at a conference have usually not been publicly available before the event and are therefore unlikely to cite each other directly. Nevertheless, several works [26]–[28] show that social information expressed by conference participants can be used. In contrast, with our approach, these works use the networks independently and do not explore how they can be combined with features extracted from the text to better find similar papers.

In our previous work [29] we used a network constructed from paper-bidding preferences expressed by the reviewers in the paper-bidding phase, where each reviewer marks the papers that he/she would like to review. In the resulting network, two papers are connected if the same reviewer expressed a preference to review them. We presented an automatic conference scheduler using this network in combination with the TF-IDF (term frequency-inverse document frequency) weighted vectors of paper abstracts. In this paper, we use several additional NLP methods such as terminology extraction and neural text embeddings to extract useful information from the text and additional graph-based data such as bibliographic coupling networks. We transform graph-based data into the vector form using both Personalized PageRank [30] and node2vec [31] algorithms and compare the results. Furthermore, we select the best features using feature-subset selection methods and extensively evaluate the individual components, as well as the final automatic scheduling algorithm as opposed to relying on expert evaluation and silhouette score used in our initial work [29].

C. DOCUMENT CLUSTERING

Several authors have shown that standard document-clustering methods, such as k-means [32] and mean shift [33]), can be improved with feature-extraction approaches that select the optimal features to be used during clustering. Abualigah [34] shows that feature selection with a modified Krill Herd algorithm can improve the k-means document clustering. Abualigah and Khader [35] and Abualigah *et al.* [36] present a similar approach using the particle-swarm optimization. The authors show that selecting

a limited subset of features can improve the results compared to using all the available features. We show that the same applies to our case.

However, the above approaches do not take into account the limitations on cluster sizes. When clustering articles to generate a conference schedule, cluster sizes have to match the sizes of the plenary and parallel sessions at the conference. Several authors propose constrained-clustering methods for conference-schedule generation that aim to solve this issue. Vallejo *et al.* [37] present two algorithms, Clustering Algorithm with Size Constraints and Linear Programming (CSCLP) and K-MedoidsSC, which could produce clusters that match a predefined conference schedule. CSCLP treats clustering as a global optimization problem, while our approach uses local optimization to ensure good clustering while satisfying the size constraints. K-MedoidsSC starts with a clustering returned by the K-Medoids algorithm and then rearranges the clusters to satisfy the size constraints. In our approach we start with the results of the K-Means algorithm, modified to satisfy the size constraints and use local optimization to improve the results after the size constraints have been satisfied. Kalmukov *et al.* [38] present a similar approach using hierarchical clustering, which does not produce clusters matching the conference schedule, but is specifically designed to cluster the conference papers. Kudo *et al.* [39] address the problem of additional constraints that can be present when constructing conference schedules, e.g., two paper presentations given by the same author cannot occur at the same time.

Unlike our approach, no existing work presents a complete system for automating conference scheduling. We present an end-to-end approach that combines feature extraction with a constrained-clustering algorithm to generate a complete conference schedule. The system is available as a web application. We present a comprehensive evaluation including synthetic and real-world datasets, real-world conference schedules and multiple languages.

III. PROPOSED AUTOMATIC CONFERENCE SCHEDULING

A conference program usually consists of workshops, tutorials, invited talks, and conference tracks. We focus on constructing an automatic schedule of the conference tracks, where accepted research papers are presented. The presentations are grouped into sessions, where each session contains presentations of the same or similar topic. In larger conferences, multiple sessions run simultaneously and will cover different topics to allow attendees to choose a topic of interest to them. An entire schedule consists of multiple sessions, usually spread across several days. In our approach we assume that the number of sessions, their duration, as well as their distribution in the schedule are defined in advance. The schedules depend on external factors, such as the number of rooms available at a venue. Our software supports the ability to create a new schedule or to reuse previously defined schedules from the same conference series. The automatic assignment of papers to sessions has to take into account that the number of papers across topics varies and might not correspond well to the available sessions. Additionally, many papers address more than one research topic.

In this section we present our approach to the automatic construction of conference schedules. In Subsection A we describe classic methods to find similar papers. We use the extraction of textual features, feature weighting, and terminology extraction. In Subsection B we describe our application of neural text embeddings, with the focus on word, sentence, and document embeddings. We then extract the information from metadata, such as citation networks and co-bidding networks, with Personalized PageRank and node2vec vectorizations (described in Subsection C). In Subsection D we describe a novel, k-means-based clustering algorithm that ensures that the returned clusters fit the predefined schedule structure and maximizes the distance between parallel sessions. Our approach takes into account that papers might belong to more than one topic and minimizes the misplacement error using a local search.

A. PAPER-SIMILARITY APPROACHES AND TEXT EMBEDDINGS

The classic approaches to text similarity describe each text with a variety of features. In our work we extract features from two types of data sources: the text content of the papers and the networks constructed from the metadata, such as citations. Fig. 1 gives an overview of the process of obtaining vector representations of papers. Although modern, neural-network-based text representations are significantly more successful at the word- and sentence-level, classic document-level approaches are still competitive [40]. We first shortly outline the bag-of-words (BoW) representation with TF-IDF weights, followed by terminology-based approaches that contain highly relevant information for domain-specific documents such as scientific papers.

1) BAG-OF-WORDS AND TF-IDF WEIGHTS

A standard approach to presenting text-based information in a sparse vector form is to use BoW document representation with TF-IDF weights. We first construct a BoW vector for each document, with a dimension equal to the vocabulary size (i.e., the vector is sparse). Each vector component consists of two parts: the term frequency $\mathrm{TF}(t, d)$ equals the frequency of the word t in the document d, and the inverse document frequency $IDF(t)$ of word t is calculated as:

$$IDF(t) = \log(\frac{N}{n_t}),$$

where N is the number of documents in the corpus and n_t is the number of documents containing the word t. The final value is calculated as $\mathrm{TF\text{-}IDF}(t, d) = \mathrm{TF}(t, d) \cdot \mathrm{IDF}(t)$. TF-IDF gives larger weights to words that only appear in a small number of documents and lower weights to words appearing in many documents. This is based on the intuition that rare words are characteristic of a certain topic. We use the

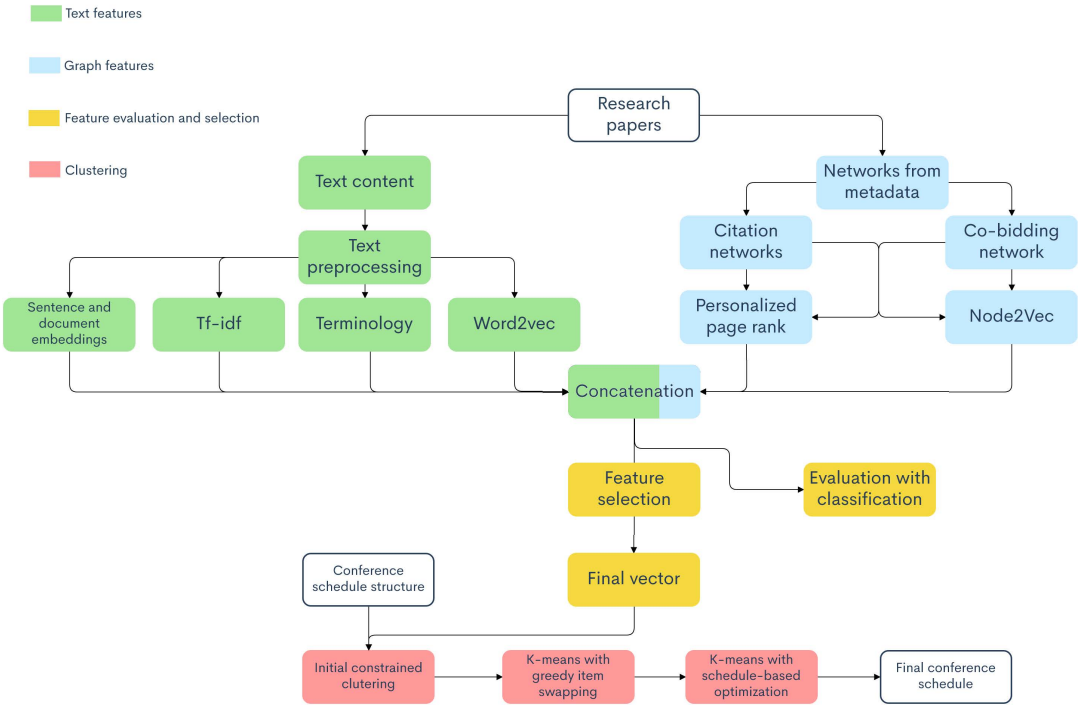

FIGURE 1. Overview of our NeSyChair automatic conference-scheduling system, consisting of text and graph representation construction, feature selection, and constrained clustering. The four main components of the system are indicated with different colors. We extract features from the text and network metadata concatenating them into a single vector. Most of the features in the vector are text-based, as shown in Table 4. The final set of features is selected using feature-selection techniques.

TF-IDF representation of documents as a baseline approach, as it usually produces reasonably good results.

A disadvantage of the BoW representation is that words are represented as independent dimensions, which means that the semantic similarities between words are not captured. For example, in the BoW representation, the words *run* and *running* are treated as different vector components even though they are semantically similar. Such information can be retained with dense word embeddings, which transform words into dense vectors in such a way that the distances between vectors describe the semantic relations between the words. We present dense word, sentence, paragraph, and document embeddings below.

2) WORD AND PARAGRAPH VECTORS

Some of the most popular word-embedding methods are the continuous bag-of-words (CBOW) and skip-grams models implemented in the word2vec tool [17]. Both approaches train a shallow neural network and extract network weights as word representation vectors. CBOW trains a neural network model to predict a word based on its preceding and succeeding words (the context), while the skip-gram model predicts the context based on the word. The weights of the trained neural network connected with a particular input word are used as its word embedding. As semantically similar words appear in similar contexts, they are assigned similar vectors that contain information about the semantic relationships between words.

While word embeddings are useful, we cannot use them directly to measure the similarity between documents (i.e., papers) since the resulting vectors represent individual words and not entire documents. We can represent a document by taking an average of all the word embeddings as the document embedding. In our case this method did not produce good results concerning document similarity, as the resulting paper vectors were all close to each other. Another approach is to produce embeddings of larger text units. One such approach is doc2vec [3], which extends the CBOW and skip-gram models to paragraphs, sentences, or entire documents (instead of words). Unlike training a model to predict the following word based on its context, they train a model to predict a word based on a unit of text of variable length, which allows them to create embeddings for larger units of text, such as sentences or paragraphs. We used this approach to construct document vectors from the full text of each paper.

3) UNIVERSAL SENTENCE EMBEDDINGS

In addition to doc2vec embeddings [3] we also used the universal sentence encoder (USE) [5] to obtain text representations. USE uses the state-of-the-art transformer neural network architecture [18], which is well-suited to obtaining contextual information from texts. Cer *et al.* [5] compute

embeddings not only for sentences, but also for paragraphs and larger units of text. First, word embeddings are obtained from the embedding layer in the transformer architecture. Then, sentence or paragraph embeddings are obtained by computing the element-wise sum of word representations at each word position. This approach was not designed for long texts such as scientific papers and gives poor results when generating a single embedding vector from the entire paper. To remedy this, we split papers into smaller, n-character-long texts. After calculating the vectors for these chunks, we obtain the vectors for full papers using one of the following two approaches:

1) the vectors for full papers are obtained by averaging these vectors, and
2) in the text-classification setting, each of these vectors is classified individually and the median prediction is taken as the final result.

The first approach was used to obtain vectors for larger text chunks with the word2vec approach [41]. The second approach is not commonly used, but is motivated by the fact that scientific papers can contain sections of text that are hard to categorize, such as equations and tables. By averaging the vectors, such sections might negatively impact the final representation vector. Since such sections rarely compose the majority of a paper, classifying each vector individually and using the median of predictions as the final result can be considered as a de-noising method. A downside of this approach is that it is not suitable for clustering, so it cannot be used when constructing a conference schedule and is therefore only useful to check whether the embeddings contain useful semantic information. We also use the second approach to classify papers using BERT embeddings.

To find the optimal length of text chunks, we tested the performance for chunks of 1024, 2048, 4096, and 8129 characters using an internal 10-fold cross-validation on the training set (see the description of our datasets in Section IV-B). We used the embeddings calculated on the chunks to classify the papers into their research topics using a manually labeled dataset. The classification accuracy of this evaluation is presented in Table 1. The results show that we obtain better results by splitting the texts into smaller chunks. Averaging the vectors works better than using the median of predictions for every chunk size. We obtained the best results with chunks of 2048 characters.

4) FINE-TUNED BERT EMBEDDINGS

There are several embedding approaches based on transformer neural network masked language models, such as BERT [2]. These language models are first pre-trained on large amounts of unlabeled text and then fine-tuned on a specific task. The approach produces state-of-the-art results on a variety of tasks, even with only a small amount of task-specific data [4]. In a classification setting, we follow the approach presented by Turc *et al.* [4]. We start with the original pre-trained BERT model [2] and append to it a single dense layer with the number of hidden neurons equal to the number of class values. We fine-tune this model on our training data using the AdamW optimizer [42]. Due to the large size of the BERT model and the memory limitations of the GPUs, we cannot produce BERT embeddings for the entire text. Therefore, we split each document into chunks of 512 tokens, which matches the length the model was pre-trained on.

We fine-tune the pre-trained model on our manually labeled paper datasets to classify the papers into their research topics. This updates the weights used in embeddings, capturing more semantic information relevant to the task. The paper datasets are described in Section IV.

We obtain the final document classification by splitting a document into chunks of 512 words, classifying each chunk, and taking the median prediction of each part. Unlike with USE, for BERT, representing the entire document with the mean vector of all the chunks produced worse results than taking the median of the prediction.

TABLE 1. The classification accuracy for topic assignment using different chunk sizes with the USE representation. The best result is in bold.

Chunk size (characters)	CA (%, mean vector)	CA (%, medians)
1024	24.5	20.0
2048	**25.0**	19.5
4096	20.7	17.2
8129	16.4	11.9

5) TERMINOLOGY EXTRACTION

One of the approaches to identifying the topics of a research paper is to use terminology extraction to identify phrases in a given text that are specific to a certain field. Ignoring words found in a general vocabulary and focusing on domain-specific words might better determine the topics of a paper. A standard terminology-extraction approach is to use language-specific rules identifying terminology candidates and use statistical approaches to select the best ones [43]. However, this requires language-specific rules and does not generalize well across languages.

Terminological expressions can be extracted using contextual word embeddings in an unsupervised manner [44], [45]. Pre-trained contextual embeddings are available for many languages and can improve the performance in many NLP tasks [2]. As they can be used for unsupervised keyword extraction, additional task-specific training is not required, making such approaches language-independent.

We use the approach of Bennani-Smires *et al.* [45] that computes contextual embeddings for individual words and multi-word phrases. It identifies phrases relevant to the topic of the document by comparing phrase embeddings to the embedding of the entire document. Words with similar embeddings to the document are selected as keywords. Additionally, the procedure ensures that selected keywords are sufficiently different from one another by measuring the distance of their vectors. This ensures that the selected keywords are suitably diverse.

We adapted the approach for the specific task of automatic scheduling. As our goal is to form small groups of related papers to match the sessions in the conference schedule, the terms that appear in too few or too many papers are not helpful. We removed the candidates that appeared in less than three papers or in over 15% of all papers. We also filtered out some frequent errors, such as the candidates containing common journal or conference names, terms containing only a single letter (for example, terms like *graph G* or *Matrix M*), and terms shorter than five letters, which appeared due to errors when converting papers from PDF to a raw text format.

We also checked whether limiting the number of extracted terms would improve the overall performance. Extracted keywords were ranked using a statistical approach described by Penas *et al.* [43], who grade the terms based on:

- the relative frequency of the candidate term in research papers, $F_r(t)$,
- the relative frequency of the candidate term in a general corpus, $F_g(t)$, and
- the number of papers the candidate term appears in, $D(t)$.

The terminology score of candidate term t is calculated as:

$$S(t) = 1 - \frac{1}{log_2(2 + \frac{F_r(t)D(t)}{F_g(t)})}$$

Score $S(t)$ decreases with the relative frequency of the terminology candidate t in a general corpus $F_g(t)$, and increases with the relative frequency of the candidate term in research papers $F_r(t)$ and the number of papers the candidate appears in ($D(t)$). The terminology candidates are ranked by $S(t)$ and we select the top-ranked candidates as terminology features. Our analysis, presented in Section IV, shows that using a small number of top-rated terms (i.e., 100) better determines the similarity of papers than using a large number of terms.

B. GRAPH-BASED FEATURES

Useful information about paper similarity is contained in the papers' metadata, which can be extracted by presenting papers in a graphical form and using network-analytic approaches. Commonly used metadata are citation networks, where nodes represent papers. Paper u is connected to paper v if u cites j. Such graphs are useful when searching for similar papers [46]. Since citations between papers presented at the same conference are rare, we construct a graph based on bibliographic coupling [22], where papers u and v are connected if they cite a common paper. We weigh the connections in this graph according to the number of citations the connected papers share. The assumption behind these connections is that papers that share citations are more similar to each other than papers that do not.

Conference organizers have access to additional metadata that can be useful in determining the similarity of papers. An example of such metadata are the preferences expressed by reviewers in the bidding phase of the paper-evaluation period, when reviewers are asked to bid for submitted papers they would prefer to review. Since reviewers prefer reviewing papers from their own field of research, the papers that the same reviewer bids for are likely to be similar. We captured reviewers' preferences by constructing a co-bidding graph [29], where two papers are connected if the same reviewer expressed a preference to review them. We weighted the connections between the papers with the number of reviewers who expressed a preference to review the connected papers.

For graph-based similarity information to be used together with other extracted features, it has to be embedded in a vector form. We used two embedding methods—Personalized PageRank [30] and node2vec [31]—and applied them to the bibliographic coupling graph and co-bidding graph.

1) PERSONALIZED PageRank (PPR)

This algorithm was originally designed to rank nodes by importance in a network of web pages. It constructs node representation using a random surfer model, where a web surfer moves through the nodes by either randomly following hyperlinks or returning to the starting set of nodes with a small probability. Using this model, the PPR algorithm calculates the probability of a surfer reaching every other node starting from a specific set of nodes. These probability distributions, calculated separately for each node in a network, represent the nodes' embeddings. We can calculate the Personalized PageRank score for every node using the following equation:

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v} + (1 - c)I(u),$$

where $R(u)$ is the probability of reaching node u, B_u is the set of all nodes pointing to node u, N_v is the number of outgoing links from a neighboring node v, and $1 - c$ is the probability of returning to the starting set of nodes. $I(u)$ is a function of the starting nodes and maps a node u to the probability of the random surfer returning to it when it returns to the starting set of nodes. For example, if we define the starting set to consist of a single node s, then we can set $I(s) = 1$ and $I(x) = 0$ if $x \neq s$.

This approach can be applied to a citation network or a network of bibliographic coupling, as shown by Grčar *et al.* [47]. We compute the PPR vector R for each paper p by setting $I(p)$ to 1 and all the other components of I to 0. This produces a vector of all the papers, where the papers close to p have higher values than the papers farther away from p. Two nodes in a network are similar if the PPR algorithm computes similar vectors for them. This is useful in the search for similar papers, as shown in our previous work [29].

2) Node2vec

A disadvantage of the Personal PageRank vectorization is that it only considers the distances between the nodes of a

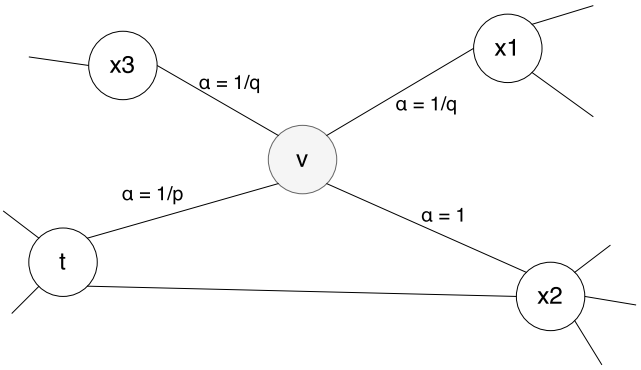

FIGURE 2. Illustration of the search bias used in node2vec. Node v is the current node and t is the previous node of the walk. Edges to t have a bias $\alpha_{pq}(t,x) = 1/p$. Edges to neighbors of t have a bias $\alpha_{pq}(t,x) = 1$. Other edges have a bias $\alpha_{pq}(t,x) = 1/q$.

network, while useful information can also be present in the shape of the network. Node2vec attempts to capture such information using a vectorization approach. It first defines the node neighborhood using a random walk. Let u be the starting node and c the sequence generated by a random walk, with $c_0 = u$. The random walk is generated using the following equation:

$$p(c_i = x | c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z} & : (v,x) \in E \\ 0 & : \text{otherwise} \end{cases}$$

where E is the set of edges, π_{vx} is the transition probability between v and x, and Z is the normalization constant such that the probabilities π_{vx} sum to 1. The probabilities π_{vx} are defined as $\pi_{vx} = \alpha_{pq}(t,x) \cdot w_{vx}$, where v is the current node of the walk, x the next node of the walk and t the previous node of the walk. In weighted networks, w_{vx} is the weight of the edge connecting v and x. In unweighted networks, $w_{vx} = 1$. Parameter $\alpha_{pq}(t,x)$ is called the walk bias and is defined as:

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & : d_{tx} = 0 \\ 1 & : d_{tx} = 1 \\ \frac{1}{q} & : d_{tx} = 2 \end{cases}$$

where d_{tx} is the unweighted distance between t and x, and p and q are parameters used to influence the behavior of the search. Fig. 2 shows a graphical representation of the search bias. When $q > 1$ the search prefers nodes close to the start; if $q < 1$, nodes further away from the starting node are preferred, and p determines how likely the search is to backtrack to already-visited nodes. With $p > \max(q, 1)$, the search will rarely backtrack. The parameters p and q allow for different search strategies and make node2vec more general than Personalized PageRank.

Using the neighborhood generated by the random walk, node2vec trains a shallow neural network to predict the neighborhood of a given node. As with word2vec, the weights of the trained neural networks are used as the vector representations of nodes.

C. CONSTRAINED CLUSTERING

After the extraction of features from the text and metadata, we use the obtained numerical representation to cluster similar papers into groups that match a predefined conference schedule. Commonly used clustering algorithms, such as k-means clustering [32] and Mean Shift [48] are not suitable for this task since we cannot specify both the number and size of the clusters. The constrained-clustering algorithms like COP k-means [49] are also not suitable since they define constraints at the level of individual instances, e.g., two instances cannot appear in the same cluster. Our approach is similar to the work of Ganganath *et al.* [50] (see Section II for differences). Our algorithm consists of two steps: i) a modification of the k-means clustering algorithm that allows constraints on the size of each cluster, and ii) additional optimization that takes into account the structure of the schedule and reduces the overlap of papers with similar topics in parallel sessions. The details of the two steps are described below.

1) INITIAL CONSTRAINED CLUSTERING

We perform the initial clustering in three steps. Let $C_1, C_2, \ldots, C_n$ be the n clusters we want the algorithm to return, and $s_1, s_2, \ldots, s_n$ be the desired sizes of these clusters. The clusters are initially empty. As with the k-means algorithm, each cluster C_i is assigned the mean μ_i. In the first step, we select the initial means using the k-means++ algorithm [51], which modifies the initial cluster selection by sampling points with a weighted probability based on the square distance to the nearest already-chosen cluster center (as opposed to doing it randomly, as is the case with the standard k-means). This initialization leads to better initial cluster centers and reduces the final error.

In the second step of the initial clustering, we compute the potential error e_i, i.e., for each paper representation x_i, we compute the difference between the distance to its closest mean μ_i^1 and distance to its *second-closest mean* μ_i^2:

$$e_i = ||x_i - \mu_i^2|| - ||x_i - \mu_i^1||.$$

The potential error e_i occurs if we fail to assign a paper to the closest cluster (represented by μ_i^1) and have to settle for the second-best choice (represented by μ_i^2). This quantity is a realistic assessment of the error and we use it in subsequent optimization. The idea for this computation comes from the margin-based loss used in some classification approaches, e.g., random forests [52]. Using the potential error to guide the paper-placement process and later local optimization is a novelty of our algorithm.

We sort the papers x_j according to the potential error e_i in descending order. Wrongly assigning papers with a large e_i produces a large error, so we prioritize their placement. We greedily assign papers to their closest clusters according to their e_i, starting with the paper with the largest e_i. If we attempt to assign a paper to a full cluster, we mark the cluster as full and recalculate e_i for all unassigned papers that use that cluster in the calculation of their e_i. We re-sort unassigned

papers and continue with the assignment process until any unassigned paper remains.

In the third step of the initial clustering, we improve the obtained cluster assignment of the papers by local optimization, i.e., we swap papers that reduce the overall clustering error. We define the error of the clustering as the sum of the distances between papers and the mean μ_i of their cluster c_i:

$$\epsilon = \sum_{i,j} ||x_j - \mu_i||$$

To reduce ϵ, we iteratively perform the following steps:

1) **Calculate new means.** Each mean μ_i is recalculated to represent the centroid of its cluster. We calculate the new means as:
$$\mu_i = \frac{1}{|c_i|} \sum_{x_j \in c_i} x_j$$
2) **Calculate distances to means.** For each paper x_j we calculate the distances to all the means μ_i:
$$D(x_j, \mu_i) = ||x_j - \mu_i||$$
3) **Sort the papers.** We sort the papers according to their potential error, i.e., the difference between the distance to their current cluster mean $D(x_j, \mu_i)$ and the distance to the closest mean outside of their cluster $D(x_j, \mu_k)$. If $D(x_j, \mu_i) - D(x_j, \mu_k)$ is larger than 0, than the paper is closer to the mean of cluster k than its current cluster. The error ϵ would decrease if the paper is reassigned to cluster k. Papers where $D(x_j, \mu_i) - D(x_j, \mu_k)$ is largest would benefit most by being assigned to another cluster, so we attempt to reassign these papers first.
4) **Swap the papers.** Since we are constrained by cluster sizes, we cannot simply reassign a paper to the better cluster k. Instead, we check whether a paper from the cluster k would benefit from being assigned to i. If such a paper exists, we swap the two papers. We swap the papers in the order computed in step 3.

We repeat steps 1–4 until no swap improves the clustering error ϵ. Since we only perform a swap if it reduces the distance of both papers to their means, the method is guaranteed to terminate. The pseudocode of the paper swapping step is presented in Algorithm 1.

Algorithm 1 Pseudocode of Paper Swapping

```
X ← list of papers sorted by D(x_i, μ_j) − D(x_i, μ_k)
for all x ∈ X do
    if D(x_i, μ_j) − D(x_i, μ_k) <= 0 then
        Remove x from X
    end if
end for
for all x ∈ X do
    μ ← mean of cluster of x
    k ← second closest cluster of x
    L ← all papers in cluster k
    if L ≠ [] then
        for all x' ∈ L do
            μ' ← mean of cluster of x'
            if (D(x, μ') < D(x, μ))&(D(x', μ) < D(x', μ')) then
                swap the cluster membership of x and x'
                break
            end if
        end for
    end if
end for
```

2) SCHEDULE-BASED CONSTRAINED CLUSTERING

In the next step we present a clustering method that takes into account the schedule of a conference. The approach described above produces clusters that have the same sizes as slots in a conference, but does not take into account which of these slots represent plenary or parallel sessions. Our goal is to generate a schedule where papers from the same topic do not occur in parallel sessions running simultaneously. We define another error function to evaluate the quality of the generated schedule with regard to that goal. If a conference has no parallel sessions, this step is skipped.

Ideally, the error function would be the total number of the same topic paper pairs assigned to two simultaneous parallel sessions. However, during clustering we do not know the exact topics of papers. Instead, we assume that the ideal cluster of a paper is the one with its mean being the closest to the paper; therefore, we define the error as the number of paper pairs belonging to the same ideal cluster, but clustered into different simultaneous parallel sessions. To calculate this error, we count the number of overlaps between each pair of clusters.

We reduce this error in two steps. First, we perform **greedy slot assignment**. We count the number of overlaps between each pair of clusters and assign clusters to slots in the following way:

1) Assign clusters with the largest overlaps into plenary sections to ensure no other session will occur at the same time.
2) Assign clusters with the largest remaining overlap to the first slot in each parallel session to ensure they will not overlap.
3) Assign the remaining clusters to the parallel session where they have the lowest overlap with other clusters already in the session.

The pseudocode of the procedure is presented in Algorithm 2.

The approach is not guaranteed to minimize the error function. The error can be further reduced by swapping the papers between clusters to reduce the overlap in parallel sessions. We treat this as another optimization problem and use a genetic algorithm to further reduce the error function. The initial clustering and greedy slot assignments are not strictly necessary. Instead, we could have optimized the final schedule using a genetic algorithm starting from a random initial solution. However, our preliminary experiments show that

Algorithm 2 Pseudocode of Greedy Slot Assignment

```
L_plenary ← list of plenary sessions
L_parallel ← list of parallel sessions
P ← list of clusters sorted by number of overlaps
for all p ∈ P do
    if L_plenary is not FULL then
        insert(L_plenary, p)
    else if L_parallel has empty first slot then
        PS ← first parallel session with an empty first slot
        insert(PS, p)
    else
        L_remaining ← all parallel sessions with empty slots
        overlaps ← []
        for all i ∈ 1 : L_remaining do
            overlaps[i] ← number of overlaps between
            papers in L_remaining[i] and p
        end for
        Sort L_remaining by the number of overlaps
        insert(L_remaining[1], p)
    end if
end for
```

better initialization leads to lower final errors and decreases the computation time, especially for large conferences.

This **schedule-based optimization** returns better schedules than the other two approaches as, besides the schedule structure, it also takes into account the parallel sessions. However, the error function used during the optimization is still an approximation of the real error function, which cannot be obtained without knowing the exact topics of the papers.

Several more complex algorithms could be used to perform the optimization (e.g., the approaches presented by Abualigah *et al.* [53] and Abualigah *et al.* [54]). However, our preliminary experiments show that the genetic algorithm used reduces the error function to almost zero in all the test cases, making the use of more advanced approaches unnecessary.

Another way to calculate the error function would be to perform topic assignment beforehand (e.g., using the LDA algorithm [55]). In principle, this would better approximate the true paper topics. However, such an approach can be problematic for broad topics with multiple sub-fields (e.g., deep learning). Such topics would need to be split into properly sized sub-topics that would fit the conference schedule, and the problem would translate from constrained-clustering to constrained-topic assignment. We leave the testing of this approach for future work.

3) COMPUTATIONAL COMPLEXITY

The most complex step in our pipeline is the initial constrained clustering approach with the time complexity of $\Omega(n^2k)$ for both the initial cluster assignments and the paper-swap optimization, where n is the number of papers and k is the number of clusters. While quadratic dependence on the number of papers is seemingly disadvantageous and makes the approach scale poorly with large numbers of papers, we note that this is not of particular practical concern. Our system is designed for scheduling scientific conferences and only few conferences publish more than 2000 papers - the number that is quite manageable by our system.

IV. RESULTS AND DISCUSSION

An objective evaluation of the created schedules is difficult. A comparison of the automatically generated schedules with the ones used in actual conferences could be misleading, since humans could produce many different acceptable schedules, even if their semantically equivalent permutations were taken into account. Besides, a dataset of actual schedules and conference papers does not exist. To overcome these problems, we first present a newly created database of research papers from the field of ML in Section IV-A. We use this database to evaluate the two main components of our approach, finding similar papers in Section IV-B and constrained clustering in IV-C. We test the approach as a whole on datasets from actual conferences in Section IV-D. In our work, we use features of different types. In Section IV-E, we analyze the impact of different feature groups to the success of clustering. Our approach is based on contextual cross-lingual representations of text and terminology. Section IV-F contains the evaluation in a multilingual setting. Finally, in Section IV-G, we comment on the implementation and reuse of the NeSyChair system.

A. DEVELOPMENT DATABASE OF MACHINE-LEARNING PAPERS

To evaluate how well we can detect papers belonging to similar research topics, we constructed a database of research papers. From this database we extracted several types of features and evaluated them in Section IV-B. We used papers from six ML conferences. Table 2 presents the number of papers from each conference. ECML-PKDD 2015 and ECML-PKDD 2016 contained more papers than those listed, but we only selected those that are publicly available.

We labeled each paper with its field of research so that we could evaluate how good our approach is at finding similar papers. For the papers from the ECML-PKDD 2015, ECML-PKDD 2016, and ICML 2016 conferences, we used the labels of the sessions the papers were presented at during the conference. In cases where a label did not appear in all three conferences, we labeled the paper with the closest field from the ICML 2016 conference. For fields with direct counterparts,

TABLE 2. The number of papers in our database by conference.

Conference	Number of papers
ECML-PKDD 2015	63
ECML-PKDD 2016	90
ICML 2016	262
SIGKDD, AISTATS, NIPS	200
Total	615

TABLE 3. Session labels from the ECML-PKDD 2015, ECML-PKDD 2016, and ICML 2016 conferences. The ICML 2016 session labels are taken as the class values for all the other conferences.

ECML-PKDD 2015	ECML-PKDD 2016	ICML 2016
Rich data	Deep learning and Neural Nets	Approximate Inference
Probabilistic and Statistical Methods	Graphs	Bandit Learning
Data Streams and Online Learning	Clustering	Bayesian non-parametrics
Sparsity and Compressed Sensing	Learning	Bayesian optimization
Matrix and Tensor Analysis	Classification	Causality
Model Evaluation and Selection	Optimization	Clustering
Classification and Supervised learning	Topic Modelling	Comp. Advertising & Soc. Sci.
Clustering and Unsupervised Learning	Patterns	Deep Learning
Graph Learning	Kernels	Deep Learning and Vision
Graphical Models and Bayesian Networks	Probabilistic learning	Deep Learning Computations
Data Pre-processing	Data science	Distributed Optimization
Pattern and Sequence Mining	Graphs and Social Networks	Feature Selection
Social and Graph Learning	Reinforcement learning	Gaussian Processes
Deep Learning	Factorization	Hashing
Regression	Streams and Time Series	Kernel Methods
Bandits	Semi supervised learning	Large Scale Learning
Large Scale Learning and Big Data	Dimensionality	Learning Theory
Preference Learning	Patterns in Sequences	Manifold learning
Multi-task, Transfer and meta-learning	Bandits and Transfer Learning	Matrix Factorization
Distance and Metric Learning	Rich data	Monte Carlo Methods
		Natural Language Processing
		Networks and Graphs
		Online Learning
		Optimization
		Privacy
		Probabilistic methods
		Ranking learning
		Reinforcement Learning
		Sparse optimization
		Sparsity
		Structured prediction
		Sub-modularity
		Supervised Learning
		Time Series Analysis
		Topic Models
		Transfer Learning
		Unsupervised Learning
		Variational Inference
		Vision

we renamed the field to the name used in the ICML 2016 conference (e.g., we renamed *bandits* to *bandit learning*). If a field did not have a direct counterpart (e.g., *Rich Data from ECML-PKDD 2015*), we discarded the field and the papers that appeared in it. Table 3 shows the session labels of those conferences. In addition, to increase the variability of the content, we manually labeled a selection of publicly available papers from the SIGKDD, AISTATS, and NIPS conferences (from 2014–2016) by using the same set of labels. We only labeled the papers with clearly defined research areas. Specifically, we selected 20 papers from each of the fields *deep learning*, *networks and graphs*, *kernel methods*, *optimization*, *probabilistic methods*, *reinforcement learning*, *time series analysis*, *supervised learning*, *topic models*, and *unsupervised learning*.

B. EVALUATING PAPER SIMILARITY WITH FEATURE SELECTION AND CLASSIFICATION

To evaluate our approach for finding semantically similar papers, we treat it as a classification problem, using the database of scientific papers described above in Section IV-A. As the database contains 615 papers belonging to various ML fields, it is large enough to cover most real, large conferences. We use it to compare several combinations of the proposed text and metadata representations and feature-selection methods.

In our classification dataset, we use session labels from the ICML 2016 conference as class values. We extracted several types of features and evaluated them in two ways:

1) By using feature-selection algorithms to identify the best features. We used three methods: analysis of variance (ANOVA), ReliefF, and mutual information (MI). ANOVA [56] measures how features differ statistically between the classes. ReliefF [57] weights features according to their ability to distinguish between near instances with different class values. MI [58] is an information-theoretic function measuring the mutual information between the feature and class random variables. In our tests, ANOVA returned the best results.
2) By training a classifier. We trained several classifiers—logistic regression (LR), random forests (RF), and a fine-tuned neural network BERT models—on different subsets of features. The classification accuracy served as a criterion to determine the most beneficial feature subset for finding similar papers.

The best features chosen by each feature-selection method were evaluated using the classification accuracy of the trained models. We evaluated the following feature groups:

- components of BoW vectors weighted with TF-IDF,
- bibliographic coupling network converted to vector form using node2vec,
- bibliographic coupling network converted to vector form using PPR,
- occurrence vectors of extracted terminology,
- co-bidding graph of reviewers converted to vector form using node2vec,
- co-bidding graph of reviewers converted to vector form using PPR,
- sentence, paragraph, and document embeddings obtained with four different models (skip-grams, doc2vec, USE, and BERT),
- all the extracted features,
- the best x features selected using feature-selection algorithms for various values of x.

Table 4 shows how many features of different types were included in the top 1000 features using different feature-selection methods. All methods selected the majority of features from the BoW representation.

We evaluated each group of features and several combinations of the best features using several classifiers. The database of research papers served as a training set to predict the research field of a given paper. We used different combinations of features and three classification algorithms: LR [59] with $l2$ norm, RF [52] with 500 trees and an unlimited maximum depth, and support vector machines (SVM) [60] with the linear kernel. Note that LR functions exactly the same as the last softmax layer of neural networks.

TABLE 4. Distribution of feature types in the top 1000 features selected using different feature-selection algorithms.

	Number of features in the top 1000			
Feature type	ANOVA	MI	ReliefF	Total #features
BoW with TF-IDF	965	865	849	32762
bibliographic coupling (node2vec)	29	128	89	128
extracted terminology	6	7	23	100
doc2vec document vectors	0	0	39	128
USE vectors	0	0	0	512
BERT sentence vectors	0	0	0	512

Additionally, we train an English BERT model following the implementation presented by Turc *et al.* [4]. We report these results separately at the end of the section.

The classification accuracy estimated with the 10-fold cross-validation is presented in Table 5. In total, the database contains papers from 39 research fields, with the default classifier achieving the classification accuracy of 6.1% (the proportion of the most frequent class). We obtained the best results with the RF classifier on the best 1500 features selected by ANOVA. Among the individual groups of features, the extracted terminology features and bibliographic coupling network features produced good results when using LR. The BoW vectors weighted with TF-IDF worked well in combination with other features and individually when using RF. The doc2vec representation performed poorly with all the algorithms. The vectors obtained with USE performed better and gave the best results when splitting the papers into smaller chunks of text while using RF.

The co-bidding features are not directly comparable with others, as we could only test them on 136 papers from the ECML-PKDD 2017 conference (we did not have access to the reviewers' biddings from the other conferences).

By using the top 1500 features, we achieved a classification accuracy of 57.3%. The classifiers were the most accurate when classifying distinct research fields, such as *kernel methods* and *topic models*. Papers from sub-fields were often wrongly assigned to a broader field, e.g., papers from *deep learning and vision* and *deep learning computations* were often assigned to *deep learning*). Another problem is that papers often contain approaches from different fields, which can lead to miss-classifications. For example, due to the popularity of deep learning, papers from various areas use deep-learning methods. Miss-classifications are illustrated in Fig. 3, which shows a confusion matrix of the RF classifier trained on the top 1500 features. The classifier was trained on 70% of the dataset and tested on the remaining 30%. The numbers off the diagonal represent miss-classifications. We can observe that many fall into similar topics.

When constructing a conference schedule, miss-classifications into similar topics can be non-problematic. A session containing papers from various sub-areas of deep learning could be acceptable, since all papers relate to deep learning. A paper that primarily deals with some other field (e.g., time-series analysis) but uses deep-learning methods could be placed in the same session because it is related to deep learning. To address this issue, we tested our approach with the top-N classification accuracy, where the predictions are considered correct if the ground-truth class is within the top-N predicted classes. In this test we used the same RF classifier used for construction of the confusion matrix. The results are shown in Table 6. The Top-2 accuracy improves to 63.8%, and the Top-5 accuracy is 78.3%.

1) FINE-TUNING BERT MODEL

Unlike with other types of features, for the BERT-based representation, we perform the classification using an additional softmax layer of a neural network [4]. We train the model for 10 epochs using a batch size of 32. Due to the relatively small size of our dataset compared to the large number of classes (645 papers split into 39 research topics), larger training times lead to overfitting. We achieve a classification accuracy of 40%, which is not competitive with the best classifiers using selected subsets of features.

TABLE 5. Classification accuracy (in %) of LR, RF, and SVM. The values in brackets are standard deviations of 10-fold cross-validation. The default classifier has a classification accuracy of 6.1%. The best results for each classifier are marked in bold.

Used features	LR	RF	SVM
BoW with TF-IDF	28.7 (7.8)	46.1 (3.7)	27.9 (8.1))
Bibl. coupling net. (node2vec)	41.3 (6.8)	40.5 (6.8)	29.4 (9.1)
Extracted terminology	40.7 (7.3)	28.1 (4.2)	12.9 (5.8)
Doc2Vec vectors	5.0 (4.9)	4.3 (3.3)	27.1 (6.8)
Co-bidding graph (node2vec)	2.85 (7.1)	29.0 (7.0)	**34.0** (7.2)
USE (full papers)	11.9 (5.8)	18.7 (4.5)	19.1 (9.5)
USE (2048 chunks, mean vector)	14.5 (7.1)	21.7 (8.5)	11.1 (5.2)
All features	37.6 (4.4)	49.0 (4.5)	29.0 (6.8))
Best 250	34.7 (6.0)	53.9 (5.6)	23.7 (5.9)
Best 500	41.4 (12.8)	54.2 (8.3)	25.5 (10.3)
Best 750	42.7 (6.9)	53.5 (4.7)	31.3 (7.1)
Best 1000	41.7 (8.8)	56.5 (5.4)	22.1 (5.5)
Best 1250	44.5 (7.1)	56.5 (4.3)	22.8 (5.7)
Best 1500	**45.7** (5.4)	**57.3** (7.4)	23.3 (4.7)
Best 2000	44.5 (9.6)	56.5 (5.1)	30.2 (8.5)
Best 5000	42.9 (6.8)	55.3 (4.3)	30.9 (8.3)
Best 10000	43.0 (9.1)	55.9 (5.6)	31.3 (7.0)

TABLE 6. Top-N classification accuracy of the RF classifier using the top 1500 features.

N	Top-N Classification Accuracy (%)
1	56.6
2	63.8
3	71.7
4	75.0
5	78.3
6	80.3
7	82.2
8	83.6
9	86.2
10	88.2

C. CLUSTERING EVALUATION ON SYNTHETIC DATA

The performance of the paper clustering depends on the used features. Ideally, papers with similar topics would have similar features. Unsurprisingly, the preliminary results showed that with our features we cannot produce perfect clusters. Errors in clustering could be a result of imperfect features or shortcomings of the clustering algorithm. To avoid this problem, we first evaluate the clustering algorithms in a controlled environment, using a synthetic dataset that mimics the distribution of features in the real datasets. For each of the M groups, representing topics in our dataset, we generate N two-dimensional points, representing papers. Each group is sampled from a two-dimensional Gaussian distribution with a mean vector μ and a covariance matrix Σ, where

$$\Sigma = \begin{bmatrix} \sigma & 0 \\ 0 & \sigma \end{bmatrix}.$$

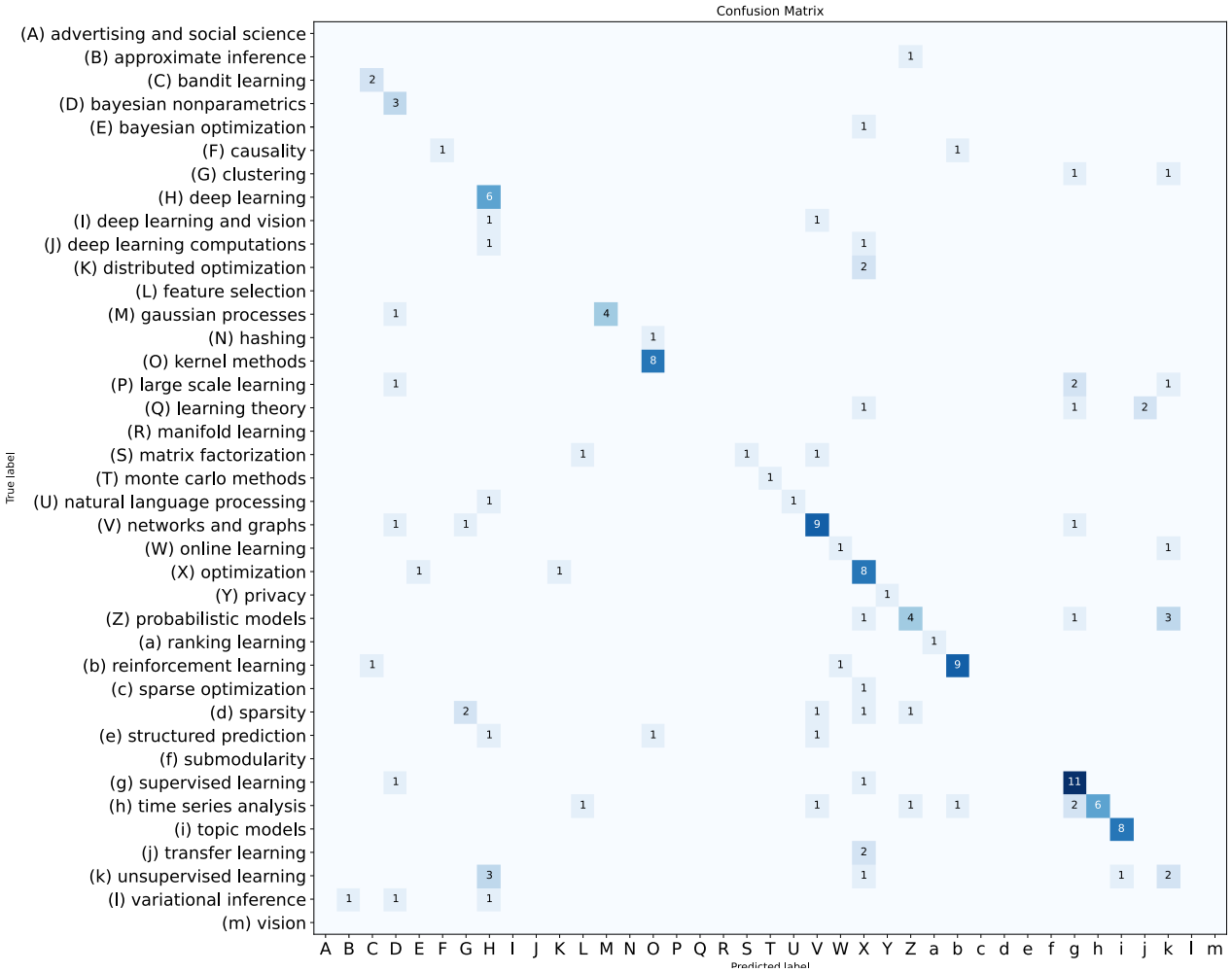

FIGURE 3. Confusion matrix of the RF classifier using the top 1500 features.

TABLE 7. The combinations of parameters used to generate the synthetic dataset for clustering evaluation.

Number of points (N)	Number of groups (M)	σ
5	5	0.10
10	8	0.15
15	10	0.20
20	15	0.25

We vary N, M, and σ of the Gaussian distributions to monitor algorithms' behaviour with different input data. The combinations used in our tests are listed in Table 7.

Fig. 4 shows a few examples of generated datasets using the settings from Table 7. The parameters were chosen so that the generated groups overlap. This makes the correct clustering challenging, but reflects the multiple topics present in the actual papers.

We compare several variations of clustering algorithms, described in Section III-C to determine which is the most suitable for our task. We evaluate the following variations:

1) random clustering, used as a baseline,
2) initial constrained clustering,
3) greedy slot assignment,
4) schedule-based optimization.

To evaluate the performance of different variants, we compute the error function counting the number of overlaps (i.e., papers with the same topic) between each pair of clusters, as described in Section III.

The goal is to reduce the number of papers from the same topic occurring simultaneously in the parallel sessions of the schedule as this would prevent conference participants from attending all the presentations from their field of interest. In the synthetic datasets, the ground-truth research areas are Gaussian distributions from which points (i.e., the papers) are generated.

Table 8 shows the errors for several clustering variants on four synthetic datasets. We report two values: i) the estimated error ϵ_{est} (i.e., using clusters to approximate topics); this error is computed in the optimization algorithm as a proxy for the actual error, and ii) the actual error ϵ_{act} computed using the ground-truth Gaussians that generated the points (i.e., topics). The actual error ϵ_{act} is not used during the optimization, as in practice it is inaccessible. We use ϵ_{est} as its proxy so there

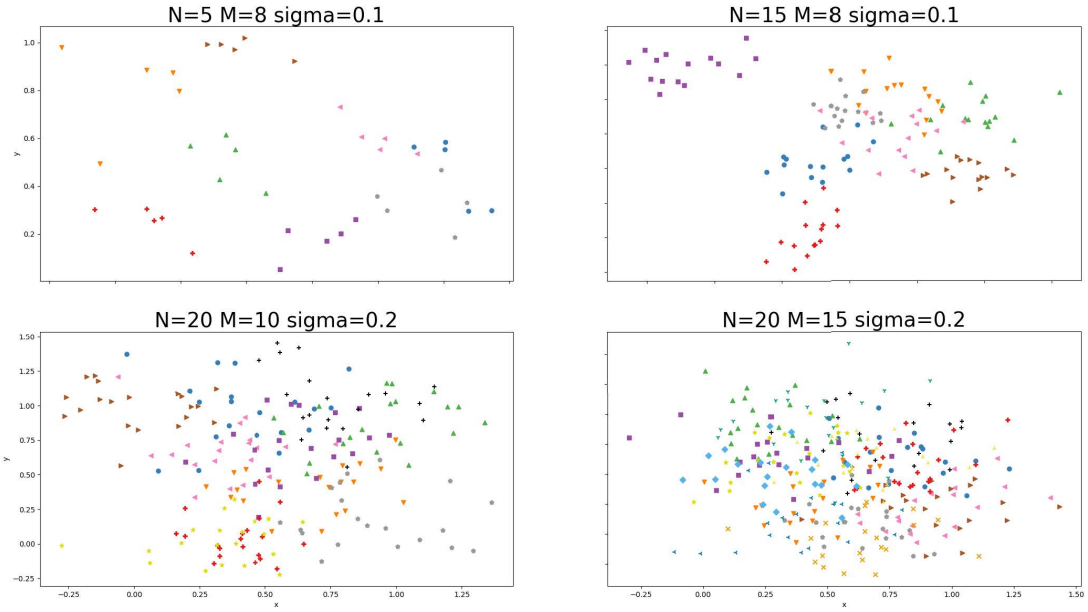

FIGURE 4. Examples of generated datasets used for the evaluation of clustering.

TABLE 8. The actual and estimated error of different clustering variants on the synthetic datasets. The dataset variants are described with (N, M, σ) triplets.

	(20, 15, 0.2)		(20, 10, 0.2)		(15, 8, 0.1)		(5, 8, 0.1)	
Variant	ϵ_{act}	ϵ_{est}	ϵ_{act}	ϵ_{est}	ϵ_{act}	ϵ_{est}	ϵ_{act}	ϵ_{est}
Random clustering	408	69	768	100	324	75	20	535
Initial constrained clustering	866	80	1224	160	531	82	26	20
Greedy slot assignment	249	31	**168**	43	159	28	17	19
Schedule-based optimization	**216**	**15**	350	**24**	**130**	**11**	**14**	**4**

is no need to manually assign the exact sub-topics to papers. For this synthetic data we can report ϵ_{act} and show how well the estimated error approximates the actual error.

The results are encouraging and show that each successive step of the proposed clustering approach in almost all cases reduces both the estimated error used in the optimization and the actual error calculated from the ground truth generating distributions. There is a strong correlation between the estimated and actual error, but the actual error is larger than the estimated error. In some cases, e.g., the (20, 10, 0.2) triplet in Fig. 8, the final optimization step increases the actual error, despite the estimated error decreasing. This happens when one or more Gaussian distributions overlap, making it difficult to determine the actual generating distribution of points. In such cases, the estimated error is an inadequate estimate of the actual error. Similar situations can occur with real-world data with papers from similar or overlapping research fields (e.g., 'machine learning' and 'deep learning'). Such situations are problematic for clustering approaches.

D. ENTIRE NeSyChair SYSTEM EVALUATION

We evaluated the NeSyChair conference scheduling system on a real-world use case, i.e., the papers accepted and presented at the ECML-PKDD 2017 conference. Because of our involvement in the organization of this conference, we have all the data and metadata available for this experimental evaluation.

TABLE 9. Structure of the ECML-PKDD 2017 paper-presentations schedule. All sessions last for 100 minutes and host 5 papers, except for the indicated sessions lasting 60 minutes and hosting 3 papers. Sessions in the same row run in parallel.

Day 1			
Session 1	Session 2	Session 3	
Session 4	Session 5	Session 6	
Session 7	Session 8	Session 9	Session 10
Day 2			
Session 11	Session 12	Session 13	
Session 14	Session 15		
Session 16 (60 min)	Session 17 (60 min)	Session 18 (60 min)	Session 19 (60 min)
Day 3			
Session 20	Session 21	Session 22	
Session 23	Session 24	Session 25	
Session 26	Session 27	Session 28	Session 29

We used the predefined structure of the conference's schedule, consisting of 136 paper presentations split into 29 sessions. The structure of the schedule is presented in Table 9. All the sessions, except four 60-minute sessions on the second day, were 100 minutes long. Each paper presentation took 20 minutes, so sessions contained either five or three presentations each.

Following the procedure described in Section III, we first construct the features, learn papers' vector representations by concatenating the representations of different feature types, and run the clustering algorithms. We use the session labels from the ECML-PKDD 2017 conference as the ground truth instead of the Gaussian distribution data. The results are shown in Table 10.

TABLE 10. Results on the actual ECML-PKDD 2017 data.

Method	ϵ_{act}	ϵ_{est}
Random clustering	376	115
Initial constrained clustering	442	346
Greedy slot assignment	68	14
Schedule-based optimization	62	0

As in the previous component testing, both the actual and estimated error are substantially reduced using greedy slot assignment and further schedule-based optimization. As before, the actual error is larger than the estimated error but they closely correlate. Again, this is due to the overlap between research fields and multiple topics contained in many papers. The low estimated error indicates that the selected features do not provide much additional information to enable improvements in topic prediction. Manual inspection of the produced schedule shows that if the NeSyChair scheduler was used to produce a draft schedule, it would already represent a sensible schedule. Using the provided user interface, the conference organizers could easily modify it into a production schedule, or interactively freeze certain parts and allow the system to fill in the rest.

While we obtained good results on ECML-PKDD 2017, a more extensive evaluation is required to determine how well the system generalizes to other conferences, especially to those outside the field of machine learning.

E. FEATURE ANALYSIS FOR CLUSTERING

When evaluating paper similarity on the classification task in Section IV-B, we show that feature selection can be used to improve results during document classification. However, it is not possible to use the same approach in clustering where ground truth labels are not available. To determine how different subsets of features affect the final clustering results, we performed feature analysis by iteratively removing sets of features from the document vectors. We split our features into four groups based on their origin: reference graphs, terminology extraction, doc2vec vectors, and bag-of-words with TF-IDF weighting. Due to the large number of features, it is possible some of them are redundant and could be dropped from the computation without harming the final results.

We evaluate the impact of feature groups on clustering by using an iterative approach, similar to the wrapper approach to feature selection. Instead of individual features as in wrapper approach, we use groups of features (mentioned above) in each iteration, thereby reducing the prohibitively large time complexity of the wrapper approach.

We first evaluate our approach on the entire set of features in the same way as in Section IV-D. We then repeat the evaluation but discard each group of features in turn. We evaluate each obtained set to find the one with the smallest error. We keep the best groups and repeat the removal process until we are left with only one feature group. The best collection of feature groups encountered during the process presents the final result. The selected groups of features obtained through iterations are presented in Table 11.

TABLE 11. Results of the feature selection during clustering.

Features	ϵ_{act}
References, terms, doc2vec, TF-IDF	62
References, terms, TF-IDF	54
References, terms	63
terms	77

As with feature selection used during the classification, the full set of features does not return optimal results. We can reduce the final error by removing the doc2vec matrix. Removing further feature groups increases the final error, though references and terms alone obtain similar results to the full feature set.

As with the whole system evaluation presented in Section IV-D, it is difficult to know how well these results generalize to other conferences, particularly outside the machine learning area. We leave this question for further work.

F. MULTILINGUAL EVALUATION

Our system does not need a labeled dataset, and uses only cross-lingual and language-independent components. This makes it suitable for use in languages other than English.

To evaluate how well NeSyChair works on languages other than English, we evaluate it on a real-world conference with parallel sessions and publicly accessible papers. We selected the Language Technologies & Digital Humanities 2016 (JTDH 2016) conference [61]. This Slovenian conference consists of 32 Slovenian research papers split into 8 sections. Additionally, the conference contains a number of student and English papers, which were not included in this evaluation. In the original schedule, English papers were presented separately from the Slovene papers, so we chose to exclude them from the clustering. The schedule contains 4 parallel slots of Slovene presentations, each containing two simultaneous sessions. Each session contained between 5 and 3 papers.

Due to its small size, JTDH 2016 does not assign specific topic names to sessions.[1] All the papers are from the field of Natural Language Processing, and no specific sub-fields are assigned to any of the sessions. However, the papers in each session still contain related papers (e.g., one of the sessions contains papers related to speech processing, while another contains papers related to digital humanities) so we use the sessions as the ground truth labels.

In assigning the papers to the schedule, we followed the approach presented in Section IV-D, but used the actual schedule structure of JTDH 2016. The results of the evaluation are presented in Table 12. As with the English conference, each step of our approach reduced the estimated error

[1] http://www.sdjt.si/wp/dogodki/konference/jtdh-2016/urnik-2016/

TABLE 12. Results on the actual JTDH 2016 data.

Method	ϵ_{act}	ϵ_{est}
Random clustering	31	52
Initial constrained clustering	43	30
Greedy slot assignment	23	5
Schedule-based optimization	23	0

and the ground-truth error. This shows that our approach can work on non-English articles as well. Both the estimated and the ground-truth errors are lower than in ECML-PKDD 2017 due to fewer papers.

G. IMPLEMENTATION AND REUSE

We implemented our approach as a web application. NeSyChair allows conference organizers to define or reuse the conference-schedule structure and use the automatic schedule to assign accepted papers in the schedule. The NeSyChair application is publicly available under a permissive license.[2]

V. CONCLUSION

The paper presents the NeSyChair system, automatically constructing conference-schedule drafts using a combination of machine learning, natural language processing, network analysis, and combinatorial optimization. The system assigns papers to a predefined schedule structure by minimizing the number of topical papers occurring simultaneously in parallel sessions. To implement the automatic scheduler, we solved two problems. We applied methods from NLP and ML to identify papers that address similar research topics, and network analysis to extract information from the available metadata. We assigned papers to the conference schedule using the modified, constrained k-means clustering algorithm that takes into account the size and the number of clusters. We further optimized the initial schedules taking into account the structure of the schedule and the overlap between papers with similar topics.

We evaluated the components of our approach on the newly constructed synthetic data and the database of research papers from six artificial intelligence and ML conferences. The entire system was used to reconstruct the schedule of the ECML-PKDD 2017 conference, using the accepted papers, metadata and the actual structure of the conference. The multilingual capability of the system was demonstrated on the Slovene JTDH 2016 conference. The developed datasets are available by request and can be used for future research.

Our evaluation was limited to ML conferences, so we cannot state how well the approach generalizes to other scientific areas. While we see no objections to its generality, a further analysis is necessary to confirm our intuition.

In practice, the NeSyChair application is not entirely automatic due to additional unexpressed preferences, e.g., participants arriving late or leaving before the end of the conference. In such cases, the produced schedule represents an excellent starting point for the organizers. The provided user interface allows for iterative schedule construction and takes manual entries into account when automatically filling in the remaining slots.

The proposed approach can be improved in several ways. It might benefit from additional features from the fields of semantic analysis and network analysis. Additional useful features could be contained in the submitted papers, such as keywords manually selected by the papers' authors. Conference organizers would benefit if the proposed automatic scheduling was integrated into existing conference-management tools, such as EasyChair.

For predicting a paper's research field, we currently only predict a single field per paper, while research papers often integrate approaches from several fields. Multi-label classification, where each paper can be labeled with an arbitrary number of research fields, might be a suitable alternative, providing useful information to conference organizers and additional features for clustering the papers into sessions. In future work, we also plan to perform a more extensive evaluation that will take into account multi-label classification. Additionally, we plan to explore whether similar approaches can be used to further improve the schedule-based constrained clustering.

[2]https://github.com/TadejSk/conference-scheduler-v2

ACKNOWLEDGMENT

The authors are grateful to general and program chairs of the ECML PKDD 2017 conference for giving us access to the accepted papers and metadata of the conference.

REFERENCES

[1] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proc. Conf. North Amer. Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Jun. 2018, pp. 2227–2237.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, 2019, pp. 4171–4186.

[3] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. 31st Int. Conf. Mach. Learn.*, vol. 32, Jan. 2014, pp. 1188–1196.

[4] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: On the importance of pre-training compact models," 2019, *arXiv:1908.08962*.

[5] D. Cer, Y. Yang, S. Kong, N. Hua, and N. Limtiaco, "Universal sentence encoder for English," in *Proc. 2018 Conf. Empirical Methods Natural Lang. Process., Syst. Demonstrations*, 2018, pp. 169–174.

[6] (2017). *EasyChair*. Accessed: Sep. 7, 2017. [Online]. Available: http://easychair.org/

[7] (2017). *OpenConf Peer-Review, Conference and Abstract Management Software System*. Accessed: Sep. 7, 2020. [Online]. Available: http://www.openconf.com/

[8] (2017). *Microsoft's Conference Management Toolkit*. Accessed: Sep. 7, 2017. [Online]. Available: https://cmt.research.microsoft.com/cmt/

[9] (2017). *IAPR Commence Conference Management System*. Accessed: Sep. 7, 2020. [Online]. Available: http://iaprcommence.sourceforge.net/

[10] (2017). *EDAS: Editor's Assistant*. Accessed: Sep. 7, 2020. [Online]. Available: https://edas.info/doc/

[11] H. Schütze, C. D. Manning, and P. Raghavan, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[12] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, Aug. 1988.

[13] G. Hurtado Martín, S. Schockaert, C. Cornelis, and H. Naessens, "Finding similar research papers using language models," in *Proc. 2nd Workshop Semantic Personalized Inf. Manage., Retr. Recommendation*, 2011, pp. 106–113.
[14] G. Hurtado Martín, S. Schockaert, C. Cornelis, and H. Naessens, "Metadata impact on research paper similarity," in *Research and Advanced Technology for Digital Libraries*. Berlin, Germany: Springer, 2010, pp. 457–460.
[15] E. Milios, Y. Zhang, B. He, and L. Dong, "Automatic term extraction and document similarity in special text corpora," in *Proc. 6th Conf. Pacific Assoc. Comput. Linguistics*, 2003, pp. 275–284.
[16] B. Jiang, E. Xun, and J. Qi, "A domain independent approach for extracting terms from research papers," in *Proc. Australas. Database Conf.*, 2015, pp. 155–166.
[17] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5998–6008.
[19] P. Sitikhu, K. Pahi, P. Thapa, and S. Shakya, "A comparison of semantic similarity methods for maximum human interpretability," in *Proc. Artif. Intell. Transforming Bus. Soc. (AITB)*, vol. 1, 2019, pp. 1–4.
[20] G. Lample, A. Conneau, M. Ranzato, L. Denoyer, and H. Jégou, "Word translation without parallel data," in *6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, Apr. 2018, pp. 1–15. [Online]. Available: https://openreview.net/forum?id=H196sainb
[21] D. J. D. S. Price, "Networks of scientific papers," *Science*, vol. 149, no. 3683, pp. 510–515, Jul. 1965.
[22] M. Kessler, "Bibliographic coupling between scientific papers," *Amer. Documentation*, vol. 14, no. 1, pp. 10–25, 1963.
[23] H. Small, "Co-citation in the scientific literature: A new measure of the relationship between two documents," *J. Amer. Soc. Inf. Sci.*, vol. 24, no. 4, pp. 265–269, 1973.
[24] C. L. Giles, K. D. Bollacker, and S. Lawrence, "CiteSeer: An automatic citation indexing system," in *Proc. 3rd ACM Conf. Digit. Libraries*, 1998, pp. 89–98.
[25] M. Hamasaki, H. Takeda, I. Ohmukai, and R. Ichise, "Scheduling support system for academic conferences based on interpersonal networks," in *Proc. ACM Hypertext*, 2004, pp. 1–4.
[26] F. Xia, N. Y. Asabere, J. J. Rodrigues, F. Basso, N. Deonauth, and W. Wang, "Socially-aware venue recommendation for conference participants," in *Proc. IEEE 10th Int. Conf. Ubiquitous Intell. Comput.*, Apr. 2013, pp. 134–141.
[27] F. Xia, N. Y. Asabere, H. Liu, N. Deonauth, and F. Li, "Folksonomy based socially-aware recommendation of scholarly papers for conference participants," in *Proc. 23rd Int. Conf. World Wide Web*, Apr. 2014, pp. 781–786.
[28] M. Cuong Pham, D. Kovachev, Y. Cao, G. M. Mbogos, and R. Klamma, "Enhancing academic event participation with context-aware and social recommendations," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, Aug. 2012, pp. 464–471.
[29] T. Škvorc, N. Lavrač, and M. Robnik-Šikonja, "Co-bidding graphs for constrained paper clustering," in *Proc. 5th Symp. Lang., Appl. Technol.*, 2016, pp. 15–27.
[30] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford InfoLab, Stanford, CA, USA, Tech. Rep. 1999-66, Nov. 1999.
[31] A. Grover and J. Leskovec, "Node2Vec: Scalable feature learning for networks," in *Proc. 22nd Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 855–864.
[32] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-means clustering algorithm," *J. Roy. Stat. Soc. C Appl. Statist.*, vol. 28, no. 1, pp. 100–108, 2010.
[33] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. Inf. Theory*, vol. IT-21, no. 1, pp. 32–40, Jan. 1975.
[34] L. M. Abualigah, *Feature Selection Enhanced Krill Herd Algorithm for Text Document Clustering*. Berlin, Germany: Springer, 2019.
[35] L. M. Abualigah and A. T. Khader, "Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering," *J. Supercomputing*, vol. 73, no. 11, pp. 4773–4795, 2017.
[36] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A new feature selection method to improve the document clustering using particle swarm optimization algorithm," *J. Comput. Sci.*, vol. 25, pp. 456–466, Mar. 2018.
[37] D. Vallejo-Huanga, P. Morillo, and C. Ferri, "Semi-supervised clustering algorithms for grouping scientific articles," *Proc. Comput. Sci.*, vol. 108, pp. 325–334, May 2017.
[38] Y. Kalmukov, "Automatic clustering of papers in thematic fields and working sessions," *Technol., Educ., Manage., Inf.*, vol. 6, no. 2, pp. 315–325, 2017.
[39] K. Kudo, "Academic meeting scheduling using an antiferromagnetic potts model," *J. Phys. Soc. Jpn.*, vol. 86, no. 7, Jul. 2017, Art. no. 075002.
[40] M. Pranjić, V. Podpečan, M. Robnik-Šikonja, and S. Pollak, "Evaluation of related news recommendations using document similarity methods," in *Proc. Conf. Lang. Technol. Digit. Hum.*, Ljubljana, Slovenia, 2020, pp. 81–86.
[41] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé, "Deep unordered composition rivals syntactic methods for text classification," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2015, pp. 1681–1691.
[42] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. 7th Int. Conf. Learn. Represent.* 2019, pp. 1–4.
[43] A. Penas, F. Verdejo, and J. Gonzalo, "Corpus-based terminology extraction applied to information access," in *Proc. Corpus Linguistics*, vol. 13, 2001, pp. 458–465.
[44] P. Sharma and Y. Li, "Self-supervised contextual keyword and keyphrase retrieval with self-labelling," Preprints, 2019, Art. no. 2019080073, doi: 10.20944/preprints201908.0073.v1.
[45] K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, and M. Jaggi, "Simple unsupervised keyphrase extraction using sentence embeddings," in *Proc. 22nd Conf. Comput. Natural Lang. Learn.*, 2018, pp. 221–229.
[46] S. Singhal and V. Pudi, "Dispersion based similarity for mining similar papers in citation network," in *Proc. IEEE Int. Conf. Data Mining Workshop (ICDMW)*, Nov. 2015, pp. 524–531.
[47] M. Grcar, N. Trdin, and N. Lavrac, "A methodology for mining document-enriched heterogeneous information networks," *Comput. J.*, vol. 56, no. 3, pp. 321–335, Mar. 2013.
[48] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.
[49] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained K-means clustering with background knowledge," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 577–584.
[50] N. Ganganath, C.-T. Cheng, and C. K. Tse, "Data clustering with cluster size constraints using a modified K-means algorithm," in *Proc. Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discovery*, Oct. 2014, pp. 158–161.
[51] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2007, pp. 1027–1035.
[52] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[53] L. M. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Comput. Methods Appl. Mech. Eng.*, vol. 376, May 2021, Art. no. 113609.
[54] L. M. Abualigah, D. Yousri, M. A. Elaziz, A. A. Ewees, M. A. Al-Qaness, and A. H. Gandomi, "Aquila optimizer: A novel meta-heuristic optimization algorithm," *Comput. Ind. Eng.*, vol. 157, Feb. 2021, Art. no. 107250.
[55] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.
[56] D. A. Freedman, *Statistical Models: Theory and Practice*. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[57] M. Robnik-Šikonja and I. Kononenko, "Theoretical and empirical analysis of ReliefF and RReliefF," *Mach. Learn.*, vol. 53, nos. 1–2, pp. 23–69, Oct. 2003.
[58] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, 2001.
[59] D. W. Hosmer, Jr., S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. Hoboken, NJ, USA: Wiley, 2013.
[60] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1997.
[61] T. Erjavec and D. Fišer, *Proceedings of the Conference on Language Technologies and Digital Humanities*. Ljubljana, Slovenia: Academic, 2016. [Online]. Available: http://www.sdjt.si/jtdh-2016/en/

TADEJ ŠKVORC is currently pursuing the Ph.D. degree with the Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia. He is employed as a Young Researcher at the Jožef Stefan Institute, Ljubljana. His research interests include natural language processing (NLP), more specifically using neural networks and word embeddings to improve the performance of NLP tools on less-resourced languages.

NADA LAVRAČ is currently a Research Councilor at the Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia. She is also a Professor of computer science at the University of Nova Gorica and the Jožef Stefan International Postgraduate School, Ljubljana, where she acts as the Head of the ICT Program. Her research interests include machine learning, data mining, text mining, knowledge management, and computational creativity. Her special interest is in supervised descriptive rule induction, where the research goal is to automatically induce rules from class-labeled data, stored either in simple tabular format or in complex relational databases. Her areas of applied research include data-mining applications in medicine, healthcare, and bio-informatics. She is the (co)author of several books, including *Foundations of Rule Learning* (Springer, 2012) and *Representation Learning: Propositionalization and Embeddings* (Springer, 2021).

MARKO ROBNIK-ŠIKONJA is currently a Professor of computer science and informatics at the Faculty of Computer and Information Science, University of Ljubljana. His research interests include machine learning, data mining, natural language processing, network analytics, and the application of data-science techniques. His most notable scientific results are from the areas of feature evaluation, ensemble learning, explainable artificial intelligence, data generation, and natural language analytics. He is the (co)author of over 200 scientific publications that were cited more than 6,000 times, and three open-source R data-mining packages. He participates in several national and international projects, regularly serves as a program committee member of top artificial intelligence and machine-learning conferences, and is an editorial board member of seven international journals.