Heliyon 9 (2023) e20003

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Research article

An automated materials and processes identification tool for material informatics using deep learning approach

M. Saef Ullah Miah [a,g], Junaida Sulaiman [a,b], Talha Bin Sarwar [a], Nur Ibrahim [c,*], Md Masuduzzaman [d], Rajan Jose [e,f]

[a] *Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, Pekan, Pahang, 26600, Malaysia*
[b] *Center for Data Science and Artificial Intelligence (Data Science Center), Universiti Malaysia Pahang, 26600, Pekan, Malaysia*
[c] *Faculty of Electrical Engineering, Telkom University, Bandung, West Java, 40257, Indonesia*
[d] *Kumoh National Institute of Technology, Gumi, 39076, Republic of Korea*
[e] *Faculty of Industrial Sciences & Technology, Universiti Malaysia Pahang, 26300, Gambang, Malaysia*
[f] *Center of Advanced Intelligent Materials, Universiti Malaysia Pahang, 26300, Kuantan, Malaysia*
[g] *Department of Computer Science, FST, American International University-Bangladesh (AIUB), 1229, Dhaka, Bangladesh*

ARTICLE INFO

Dataset link: https://data.mendeley.com/datasets/s3st6n77pr

Keywords:
Materials discovery
Process discovery
Materials 4.0
Material informatics
Entity-value extraction
Knowledge graph
EDLC

ABSTRACT

This article reports a tool that enables Materials Informatics, termed as MatRec, via a deep learning approach. The tool captures data, makes appropriate domain suggestions, extracts various entities such as materials and processes, and helps to establish entity-value relationships. This tool uses keyword extraction, a document similarity index to suggest relevant documents, and a deep learning approach employing Bi-LSTM for entity extraction. For example, materials and processes for electrical charge storage under an electric double layer capacitor (EDLC) mechanism are demonstrated herewith. A knowledge graph approach finds and visualizes different latent knowledge sets from the processed information. The MatRec received an F1 score of ~96% for entity extraction, ~83% for material-value relationship extraction, and ~87% for process-value relationship extraction, respectively. The proposed MatRec could be extended to solve material selection issues for various applications and could be an excellent tool for academia and industry.

1. Introduction

Research on energy storage devices is on the rise owing to the efforts to decarbonize major emitting sectors such as transportation and production. For example, the google scholar index retrieved over 375,000 articles on 2 Jan 2022 using the keyword "energy storage" [1], signifying the importance of the electronic economy. The electron economy is defined as electrifying the products and services and is expected to largely contribute to the efforts to achieve a net zero emission by 2050 [2]. There are two major classes of energy storage devices currently, viz. the secondary batteries (such as lithium-ion batteries, sodium-ion batteries, etc.) and electrochemical capacitors (ECs; synonymously electric double layer capacitors (EDLCs), supercapacitors, etc.). The secondary batteries offer the storage of a higher amount of energy (charge), but the charging rate (power capability) is relatively poor; besides, they are mostly developed from non-renewable materials involving high-carbon processing. On the other hand, EDLCs are developed

* Corresponding author at: Faculty of Electrical Engineering, Telkom University, Bandung, West Java, 40257, Indonesia.
E-mail addresses: md.saefullah@gmail.com (M.S.U. Miah), junaida@ump.edu.my (J. Sulaiman), talhasarwar40@gmail.com (T.B. Sarwar), nuribrahim@telkomuniversity.ac.id (N. Ibrahim), masud.prince@kumoh.ac.kr (M. Masuduzzaman), rjose@ump.edu.my (R. Jose).

https://doi.org/10.1016/j.heliyon.2023.e20003
Received 29 November 2022; Received in revised form 6 September 2023; Accepted 7 September 2023

Available online 14 September 2023
2405-8440/

from carbon materials (which can be obtained from renewable sources such as biomass) with little or no emissions and offer orders of magnitude higher power capability but with much lower storage capability than secondary batteries. Besides, the EDLCs can be recharged for 10^5 cycles compared to $< 10^3$ cycles of secondary batteries. Therefore, contemporary research is to enhance the energy storage capability of EDLCs by appropriate selection of materials and materials design [3,4]. In EDLCs, energy storage is accomplished by the adsorption of ions from an electrolyte over an electrode surface [5]; thousands of carbon materials processed from varied sources and synthetic procedures have been examined for their use as an EDLC electrode [6].

Data-driven materials discovery (also known as Materials Informatics, Materials 4.0 etc.) is currently gaining significant attention as a unique tool to minimize the materials discovery – manufacturing time gaps and has demonstrated its usefulness on several occasions; recent reviews on this topic can be found elsewhere [7–9]. Material informatics identifies relevant details from articles available in machine-readable form and extracts processes and properties of materials for a certain situation, besides storing the material data. In addition, materials informatics can provide an interface to find the relationship between different materials, processes, and properties. Several databases provide chemical entities extracted from the scientific literature [10–12] that contain only chemical and drug entities. Several other database systems based on materials science literature provide static processes and materials for inorganic materials, solid oxide fuel cells, heating, milling, and mixing processes [13–15].

Most existing databases can extract chemical entities and chemical synthesis processes from the scientific literature and are static, requiring regular updates [16]. There are few datasets and databases for any area of interest, but general chemical data and materials and processes are used in materials science. For example, no dataset exists for the materials and processes used in energy storage devices. In addition, there are very few automated information systems in this field. To support the fourth industrial revolution (IR 4.0) [17], and materials 4.0 [18], the need for such systems has greatly increased. A standard pipeline of operations is required to create such information systems, including a data collection system, a relevant item suggestion system, an entity identification system, an entity-value relationship identification system, a result retrieval system, and a visualization system [16]. Among the existing systems, very few have such a pipeline. Therefore, much work is still needed in this research area.

Numerous research has been conducted recently to explore the most important findings of a scientific paper. Material science is one of the most popular fields where many studies have been conducted to gain knowledge from scientific articles. These insights gained from articles include the identification of chemical structures; chemical reactions; chemical recipes; the discovery of materials; the discovery of synthesis processes, and the identification and characterization of material properties. Different studies use different approaches to determine these findings. Table 1 shows a comparative study of different systems and studies. The details of these systems can be found in Section 1: Related Studies in the Supplementary Information.

This paper proposes a tool called MatRec that processes data from PDF files and identifies relevant articles for the EDLC domain. While there are various sources, such as websites, Microsoft Word, and other text streams, this tool only considers PDF files as they are widely used and accepted. Website scraping is sometimes illegal, and there are also paywall barriers to web content. These are additional reasons for choosing PDF as the primary article format. MatRec captures scientific literature in PDF format and processes it into plain text using machine learning-based text extraction tools grobid [22] and Spacy [23]. After data acquisition, this tool identifies relevant documents or texts according to the specified domain using keyword extraction techniques and text similarity algorithms. Within the relevant documents, it then identifies and extracts entities (materials and processes), also known as Named Entity Recognition (NER) [24] by using a deep neural network model (also known as Deep Learning) [25]. This deep learning model was trained using a finite number of annotated articles provided by the domain experts. After entity extraction, the relationships between entity and value are also extracted using a rule-based approach. Finally, with the extracted knowledge, the tool creates a knowledge map based on a knowledge graph. In this knowledge graph, various relationships between materials, processes, values, and metadata can be found and checked for any relationships between them. In addition, a public web application has been developed for this system to display the processed information and perform the basic searching and sorting operations for processed materials, methods, and values.

The remainder of the paper is arranged as follows. Section 2, describes the proposed methodology, which includes data collection, dataset preparation, selection of relevant documents, description of the Deep Learning model development process, and knowledge graph construction process. Section 3 presents the experimental results and discussion on the results, and section 4 concludes the paper with directions for the future.

2. Methodology

The proposed method consists of data acquisition and processing; data annotation for named entity recognition; selection of appropriate papers; development of a deep neural network model for named entity recognition; identification of material and process entities from text; extraction of material-value relationships and process-value relationships; and knowledge graph generation. Fig. 1 depicts an overview of the proposed method in five stages. Phase 1 contains the data acquisition and processing tasks. In this phase, scientific articles are collected from different sources and converted to plain text with proper sentence boundaries. This task has been discussed in [26]. In phase 2, relevant articles are identified using keyword extraction, similarity index, and word embedding techniques. Phase 3 contains the Named Entity Recognition (NER) model design and development utilizing deep neural network techniques, and phase 4 consists of entity-value relation identification and extraction using a rule-based approach. Phase 5 contains the knowledge representation using the knowledge graph approach.

Table 1
Comparative study of different systems.

System	*Advantages*	*Limitations*	*Ref.*
ChemData Extractor	• Extracts chemical entities and relevant properties.	• Does not extract materials and processes from the materials science domain. • Provides a static database of entities. • Uses a rule-based approach to extract information.	[10]
ChemSpot	• Contains chemical entity names, abbreviations, drugs, molecular formulas, and IUPAC chemistry entities.	• Cannot extract materials and processes for materials science domain. • Provides a static database of entities.	[11]
OSCAR	• The database is created by extracting named chemistry entities from chemistry publications.	• This database does not include materials science publications. • Provides a static database of entities.	[12]
MatScholar	• This system extracts material, phase, application, property, and synthesis method for the materials science domain. • Uses neural network for named entity recognition task.	• Used a large number of abstracts for system training. • Used multiple models to identify relevant abstracts and named entity recognition tasks. • Entity identification is limited to inorganic materials only.	[19]
MatScIE	• This system extracts material names, methods, parameters, codes, and structures from scientific literature. • This system employed an LSTM-based neural network for the NER task.	• Uses multiple machine learning models but achieves lower F1 scores than existing systems. • Uses a comparatively large amount of training data. • Extract named entities for specific materials rather than all available materials and processes. • Does not find material and process relevant values.	[20]
Mat2Vec	• Unsupervised word embedding method can be used to find similar meaning materials and properties to user-provided materials.	• Need to train with a large amount of data. • Does not provide any database of materials or processes. Rather, it provides a dictionary of similar types of entities for any provided entity.	[21]
Synthesis recipes dataset	• Contains information about the target materials, starting compounds, synthesis methods and their conditions, and a balanced chemical equation of the synthesis reaction. • Bi-LSTM-based neural network can be used.	• Contains only synthesis recipes for inorganic materials. • Does not support PDF documents; it only extracts entities from HTML or XML documents. • Uses multiple machine learning models to identify relevant paragraphs, including topic modeling and random forest classifier. • Only works for fixed numbers of synthesis processes. • The word embedding model used in this study is trained using a large training data.	[13]

2.1. Data collection and processing (phase 1)

In the data collection and processing phase, the data relevant to the system is acquired and processed. This step is divided into several smaller steps, which are explained below.

2.1.1. Scientific paper collection

This step collects scientific papers from the Web of Science database. The Web of Science database is accessed through the EZproxy page of Universiti Malaysia Pahang [27]. Scientific papers are searched for the string '"supercapacitors" AND Carbon'. The search is filtered by year, starting from 2015 to 2020, and the results of the search are sorted from the highest citation counts to the lowest citation counts. Then the 250 most cited articles are selected and downloaded in portable document format (PDF).

2.1.2. Plain text conversion

To further process the collected scientific documents, they need to be converted from PDF to plain text. This is a two-phase process. In the first phase, Grobid [22] software converts the PDF documents to tei XML format. The conversion can be customized to process different parts of the document, such as only the text, references, or document structure. Using full-text processing, all sections of the document are processed. In the second phase, a custom tei XML parser is developed using Python programming to

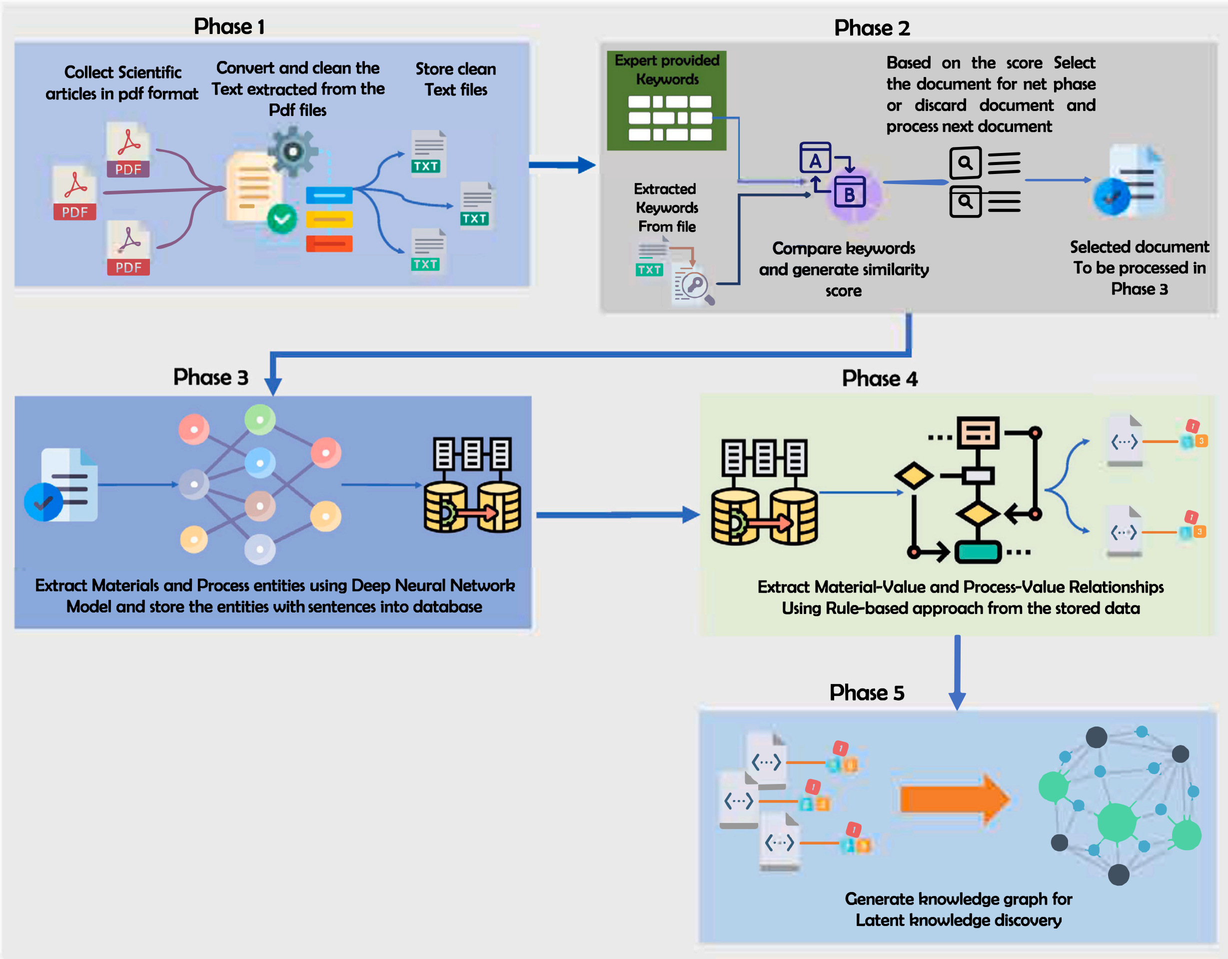

Fig. 1. Overview of the proposed tool called MatRec.

extract plain text from the tei XML format. The parser is customized to exclude certain sections, such as references, figures, tables, and acknowledgments, and extract plain text from all other document sections. The output of this phase is a plain text file in .txt format, which is saved for further processing. All the collected scientific documents are processed in this way.

2.1.3. Plain text data cleaning

In this step, various rules are applied to the plain text obtained from the previous step to extract only relevant content and eliminate irrelevant information. The PDF-to-text conversion tool discards figures and tables, while regular expressions and a rule-based approach remove reference numbers, special characters, extra spaces, and new lines from the plain text. The resulting cleaned text is then saved for further processing. Additionally, this step corrects some incorrectly converted characters, such as the degree and Angstrom symbols. The rules for text cleaning are described in Algorithm 1 systematically, outlining the steps implemented to clean up the text content produced by the PDF-to-text conversion tool.

2.2. Dataset creation for the named entity recognition deep neural network model

Of the 250 scientific documents collected, 50 documents are selected by domain experts to be annotated. This dataset is used to train the named entity recognition model. After selecting the documents, they are all annotated for two primary classes: a) Material and b) Process. For annotation, the Inside-Outside-Beginning (IOB) [28] format is used, a very common format for named entity recognition tasks. In this format, single and multiple words representing a single entity can be easily tagged. For example, a single word named “carbon” can be marked as “B-material”, and the word “polyaniline cellulose nickel” can be marked as “B-material I-material I-material”. Here, B-material represents the beginning of a material entity, and I-material represents the continuation of a material entity. Except for the classes marked “Material” and “Process”, all words are marked with an “O”, indicating that they are not part of a class. Thus, using the IOB format, five classes are tagged for all 50 documents. The tagged classes are: *a*) B-material *b*) I-material *c*) B-process *d*) I-process *e*) O.

Algorithm 1: Cleaning text content.

Input: Text String: TS
Output: Cleaned Text String: CTS
TS = Remove extra spaces
TS = Remove tab and new line characters
TS = Substitute Minus sign ($-$) hyphen sign (-)
TS = Substitute Decimal points (.) with hash sign (#) within the numbers
TS = Remove in-text reference numbers formatted with square brackets
TS = Remove in-text reference numbers with comma sign (,) and hyphen (-) signs
TS = Remove Open and Close parentheses
TS = Remove Open and Close square brackets
TS = Remove extra Comma (,), Semicolon (;), Tilde (∼), Apostrophe ('), Colon (:), Hyphen (-), Underscore (_), Forward slash (/), At (@), Quotes (' ',“ ") signs
TS = Replace “À" symbol with hyphen (-)
TS = Replace “Á" symbol with “$\sigma\eta$" symbol
TS = Replace dot (“•") symbol with degree (“°") symbol
$CTS = TS$
return CTS

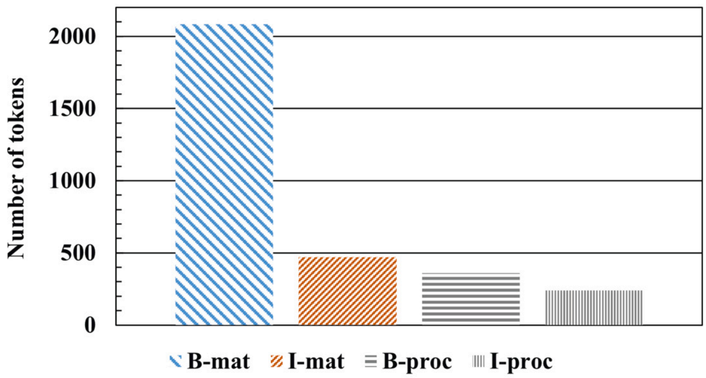

Fig. 2. Label distribution of the 4 classes of the annotated dataset. B-mat represents the words that fall under the beginning of the material class, and I-mat represents the continuation of all the words in the material class. In the same way, B-proc represents the words that fall under the beginning of the process class, and I-proc represents the continuation of any word in the process class.

The annotation process uses the UBIAI Annotation Tool [29]. The selected papers are converted to plain text and uploaded to the UBIAI web app. In the web app, a word or set of words for a single class is labeled with the appropriate class tag, and the same word or set of words is automatically labeled with the defined class. After the labeling is complete, the dataset is saved in various formats, such as a comma-separated values (csv) file and a javascript object notation (JSON) file to train different models. The entire document text is used for annotation, except for the references and acknowledgments sections. Following the annotation process, 6250 sentences were discovered to contain the word(s) marked as “material” or “process”. These 6250 sentences contain 2555 words marked ‘material’ and 600 words marked ‘process’. B-Material & I-Material classes together are considered as the ‘Material’ category, and B-Process & I-Process classes together are considered as the ‘Process’ class. Fig. 2 shows the label distribution of four classes for all 50 documents after annotation. There are 2085 tokens marked as “B-mat” and 470 tokens marked as “I-mat”. Together they give 2555 marked tokens for the entity “Material”. On the other hand, there are 360 tokens marked as “B-proc” and 240 tokens marked as “I-proc”, and together these markings give 600 marked tokens for the class “Process”. All occurrences of the words or word sequences tagged with material and processes from each document are considered, even if they are the same in different documents. The tagged words or word sequences are the same in different documents, but their container sentences are not the same and each sentence structure may have a different meaning. Therefore, when creating the dataset for the Named Entity Recognition task, all occurrences of the words are considered, whether they are the same or not.

2.2.1. Human association

Two subject matter specialists annotated the whole body of academic literature from which the dataset was compiled. Two more subject-matter specialists manually examined the produced dataset. To manually validate this dataset, the inter-annotator agreement (IAA) [30], a validation metric, is measured during the annotation process. The IAA shows how the annotation standards are clear, how consistently the annotators comprehend them, and how reproducible the annotation job is. This is a crucial aspect of an annotation task since the findings of any classification work must be verified and repeatable. To evaluate how well two raters agree on categorizing data into certain categories or labels, as well as how well a novel technique agrees with an established one, Cohen's Kappa measure is frequently utilized. The level of agreement between the two human raters in identifying particular categories or labels is measured in this study using Cohen's kappa [31]. The Cohen's Kappa score interpretation is shown in Table 2. Cohen's Kappa score increases with increased agreement between the two raters. The IAA analysis yielded a validation score, or Cohen's Kappa score, of.932 or 93%, concluding that the two annotators had the greatest agreement in accurately identifying the classes.

Table 2
Interpretation of Cohen's Kappa score.

Cohen's Kappa score	*Level of agreement*
< 0.20	Poor
0.21 - 0.40	Fair
0.41 - 0.60	Moderate
0.61 - 0.80	Good
0.81 - 1.00	Perfect

Table 3
Domain expert provided keywords for EDLC domain.

EDLC domain related keywords provided by domain experts
Supercapacitors, SCs, Electrochemical capacitors, Energy storage device, Electric double layer capacitor, EDLC, Pseudocapacitance, Electrostatic adsorption, Electrosorption, Faradaic redox reactions, Stern layer, Helmholtz double layer, Double layer formation, Activated carbon, Porous carbon, Carbon nanotubes, Graphene, Graphite oxide, GO, Reduced graphite oxide, rGO, Surface charge accumulation, High power applications, Charge separation at electrode interface, Charge separation at electrolyte interface, Non-faradaic process, Specific surface area, Pore size distribution, Electrochemical interface, EDLC characteristics, Diffuse double layer, Polarizable capacitor electrode

2.3. Appropriate paper selection (phase 2)

In the field of materials science and electric double layer capacitor (EDLC), a vast amount of scientific literature is available. However, not all documents obtained from digital library searches are relevant to the intended purpose. Hence, it is crucial to identify and select the relevant documents. For instance, this study requires scientific documents that pertain to the EDLC technology, including information about the manufacturing process, required materials and processes, and research activities. To accomplish this, a list of primary keywords is created by domain experts, which consists of keywords that are present in relevant scientific documents. Table 3 shows the list of thirty-two keywords provided by the experts, along with ten documents marked as appropriate by the domain experts. The task is to evaluate the similarity between the documents and the list of keywords extracted from them and based on the threshold value established using this procedure, select the documents for further processing. The candidate documents' keywords are compared to the provided list using similarity indices, and documents that meet or exceed the threshold value are selected for further processing to extract relevant materials, processes, and values. Fig. 3 provides an overview of selecting appropriate documents.

The selection of appropriate papers involves two sub-processes: data collection and processing, and similarity calculation. In the data collection and processing phase, relevant data is gathered and processed, after which similarity values are computed to determine which papers meet the selection criteria.

2.3.1. Data collection and processing

In this phase, the domain experts initially create a list of keywords along with a set of ten scientific documents that contain these keywords and are defined as appropriate documents. Next, the collected data is processed, starting with the conversion of PDF files to plain text using the procedure described in section 2.1.2. The plain text data is then cleaned using the method outlined in section 2.1.3. Subsequently, the processed texts are separated into two groups: positive texts and the entire text of the document. The positive texts are segmented using the grammar rules for negatives and negation [32–34].

2.3.2. Similarity calculation

To calculate the similarity between the keywords collected by the domain experts and the extracted keywords from the documents provided by the domain experts, the keywords are first extracted from the documents and then the similarity is calculated using the indices Jaccard similarity [35], Cosine similarity [36] and Cosine with word vector similarity [37].

To extract the keywords from the documents, six prominent supervised and unsupervised keyword extraction algorithms are used. KEA [38] and WINGNUS [39] are supervised techniques and for unsupervised techniques YAKE [40], TopicRank [41], MultipartiteRank [42] and KPMiner [43] are used. After the keywords are extracted from both text sets using the algorithms, they are checked for their similarity value using all the similarity indices with the keywords provided by the domain experts. The values are stored for each processed text set for each algorithm to analyze the experimental results. The similarity calculation process described above can be expressed by the Algorithm 2.

2.4. Named Entity Recognition Deep Neural Network (DNN) model development (phase 3)

The proposed approach for named entity recognition utilizes the sequence labeling technique to identify the desired named entity from a given sequence of words or sentences, by determining the class of each token. The sequence labeling task is processed by

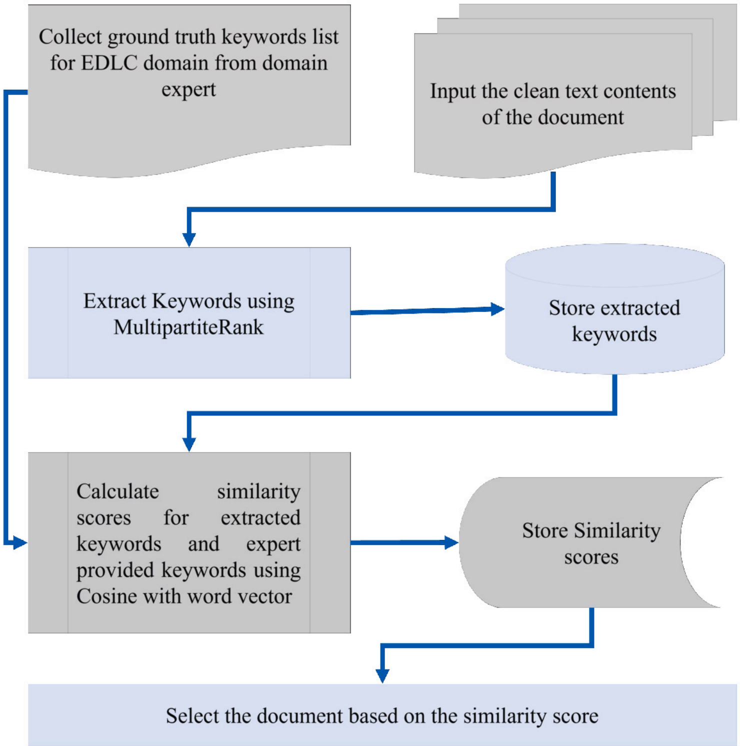

Fig. 3. Overview of appropriate paper selection procedure.

Algorithm 2: Similarity score calculation.

```
Input: Whole text String A_string
Input: Positive sentence String P_string
Input: Domain expert curated keywords list's string KW_string
Output: String containing filename, algorithm, and score
Def get_score(Sim_algo,KPalgo_name, text_content, KW_string):
    score = 0
    algo_list = ["yake","topicrank","multipartiterank",
    "kpminer","kea","wingnus"]
    if KPalgo_name in algo_list then
        algo_name = KPalgo_name keyWords = Extract Keywords using algo_name algorithm from text_content
        SimScore = Calculate similarity of keyWords with KW_string using Sim_algo
        score = SimScore
        return score
    end
    else
        return error_msg
    end
Def main(Kw_args):
    sim_algo = [jaccard, cosine, coswv]
    algorithm_list = ["yake", "topicrank", "multipartiterank", "kpminer", "kea", "wingnus"]
    for algo in sim_algo do
        for algorithm in algorithm_list do
            score_a = get_score(algo,algorithm, A_string, KW_string)
            score_p = get_score(algo,algorithm, P_string, KW_string)
            r_string = algo + algorithm + score_a + score_p
        end
        return r_string
    end
```

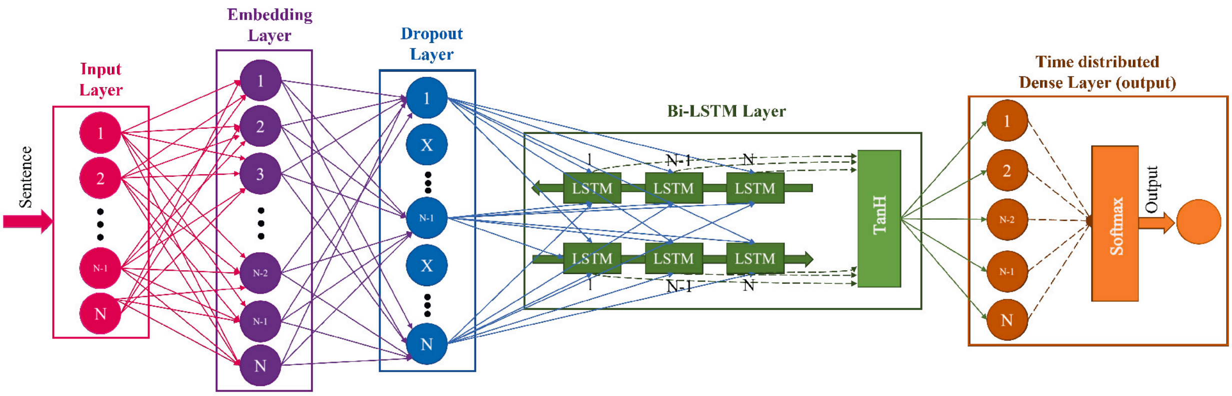

Fig. 4. The deep neural network model architecture used in this study for named entity recognition.

leveraging the word form, context within the sentence, and word representation. To perform this task, a variant of Recurrent Neural Network (RNN), specifically Long Short Term Memory (LSTM), is employed in this work. LSTM is preferred over RNN due to its ability to alleviate context problems that arise with longer sequences. A bidirectional LSTM (Bi-LSTM) layer is used in conjunction with other layers of the deep neural network model. Bi-LSTM involves the concatenation of forward and backward embeddings of input words to obtain the most optimal context of the word within a sentence. Each LSTM layer comprises an input gate (I_t), a forward gate (F_t), and an output gate (O_t), which are computed using the equations (1), (2) and (3).

$$I_t = \sigma(W_I[H_{t-1}, X_t] + B_I) \tag{1}$$

$$F_t = \sigma(W_F[H_{t-1}, X_t] + B_F) \tag{2}$$

$$O_t = \sigma(W_O[H_{t-1}, X_t] + B_O) \tag{3}$$

Here W and B represent the weight and bias of the neurons of the respective gate (X). H_{t-1} represents the output of the previous cell at time $t-1$ and X_t represents the input at the current time t. σ represents the sigmoid function.

Using the equations (1), (2), and (3), we obtain the candidate cell (Cd_t), the final cell (C_t), and the final output (H_t), represented by the equations (4), (5), and (6).

$$Cd_t = \tau(W_Cd[H_{t-1}, X_t] + B_Cd) \tag{4}$$

$$C_t = F_t * C_{t-1} + I_t * Cd_t \tag{5}$$

$$H_t = O_t * \tau(C_t) \tag{6}$$

Here τ stands for the Tanh activation function, t for the current time state, and $t-1$ for the previous time step for the respective cell and gates.

The architecture of the deep neural network model used for named entity recognition and the identification task in this study is shown in Fig. 4.

The proposed matRec model architecture is presented in Fig. 4. This model comprises five layers, including two input and output layers and three hidden layers. The first layer of the model is the input layer that receives a sentence, also referred to as a sequence of words, as input and passes it to the subsequent layer. The next layer is the first hidden layer of the model, which is a domain-specific integer-coded embedding layer. This layer is employed to reduce the dimensionality of the feature space of the sequences and facilitate the network in comprehending the contextual meaning of words. As a result, similar words have similar embedding weights for the specific domain. This layer transforms each word into a fixed-length vector for a given sentence input. To use the embedding layer, all data is integer-coded such that each word is represented by a unique integer number. A sentence sequencer method is developed to obtain all the words of each sentence, and then the words are encoded in integers using Keras Tokenizer. The embedding layer is initially randomly weighted and initiated with an embedding weight for each word in the training sentence. This layer can update the embedding weights using self-learned weights in each iteration of the training phase. The input dimension of this layer is the total data size, which is 7715 for the EDLC domain. The output dimension is the length of the vector for each word, and the input length is the maximum length of a sequence, which is determined by examining the sentence length from the dataset.

The subsequent layer in the proposed model is a spatial dropout layer, which is implemented as a regularization technique to reduce over-fitting and enhance model performance. This layer offers an efficient and cost-effective solution to mitigate over-fitting instead of utilizing ensemble neural networks. Since the training dataset used in this work is relatively small, it is crucial to reduce over-fitting. The dropout layer regularizes the network during the training phase by randomly excluding the activation and weight updates of the input and recurrent connections to the LSTM units. In this study, the Keras SpatialDropout1D layer is employed as the dropout layer. This layer eliminates complete 1D feature maps rather than individual elements, which is useful when consecutive frames in feature maps are highly correlated, and the regular dropout strategy fails to regularize the activations, thereby reducing the effective learning rate. Consequently, the SpatialDropout1D layer enhances the independence of feature maps. The input to this layer is the embedding, and the dropout rate is set to 0.2 as part of hyperparameter tuning after several adjustments.

Subsequent to the dropout layer, a bidirectional LSTM layer is incorporated into the proposed model. The bidirectional wrapper of Keras introduces two LSTM layers: one that learns the word order of the input sentence and the other that learns the reverse order of the first model. The merge_mode used for the LSTM layers is concat. TanH is the activation function while Sigmoid is the recurrent_activation function used in the LSTM layers. The LSTM layer comprises 200 neurons with a recurrent dropout rate of 0.2.

Upon completion of the LSTM layer, a time-distributed wrapped dense layer consisting of five units is introduced to classify the data into five distinct classes. The time-distributed wrapper is utilized to apply a layer to each temporal slice of the input data in order to establish a one-to-one relationship between the input and output. The dense layer is a fully connected layer typically used in the final stages of neural network construction to refine the output of the previous layer and improve the model's ability to establish relationships between the data. The softmax activation function is applied to the dense layer to normalize the network's output to a probability distribution across the possible output classes. This layer produces the final output of the deep neural network.

The proposed deep neural network model is optimized using the Adam optimizer to address the issues related to sparse gradients and noise. The categorical cross-entropy function is selected as the loss function for the model. An early stopping technique is employed to avoid overfitting the model, where the training process is halted when the validation loss reaches its minimum value. A sentence sequencer is developed to extract the sentence sequence from the labeled dataset. Subsequently, 80% of the sentences are utilized to train the deep neural network model, while the remaining 20% is used to test the model. After the training and testing phase, the model is stored and evaluated using various metrics.

The proposed deep neural network model is built using the Python programming language [44] on top of the Keras library [45] and trained using the Tensorflow library [46].

2.4.1. Materials and processes entity identification

To identify materials and processes from each provided document that has passed the appropriate paper selection stage, the cleaned plaintext is first segmented into sentences. Spacy sentencizer [47] is utilized to identify sentences due to its superior performance compared to other NLP frameworks, namely NLTK [48] and Gensim [49]. Once the sentences are segmented, each sentence is input to the prediction model which returns a list of words with a predictive class based on the training it performed on the training dataset. Only words classified as material and process are considered for further analysis. As the model output is in IOB format, each token is checked for the correct order of the IOB tags. Sequentially, if a "B-material" tag is followed by an "I-material" or "O" tag, this token marked with "B-material" is stored. The process is repeated for other tag classes. In summary, to identify material and process classes, a sequential check is performed on the following sequences of one or more words.

- "B-material" + "O" or "B-process" or "I-process", for single-worded material class. Token before "O" or "B-process" or "I-process" tag is stored.
- "B-material" + "I-material" + ... + "O" or "B-process" or "I-process", for multi-worded material class. Tokens before "O" or "B-process" or "I-process" tags are stored.
- "B-process" + "O" or "B-material" or "I-material", for single-worded process class. Token before "O" or "B-material" or "I-material" tag is stored.
- "B-process" + "I-process" + ... + "O" or "B-material" or "I-material", for multi-worded process class. Tokens before "O" or "B-material" or "I-material" tags are stored.

In the proposed method, all "I-" tags are examined to determine if there is a corresponding "B-" tag class immediately preceding it. In cases where there is no match, the "I-" tag classes are disregarded. For instance, if a token with "I-material" is detected at the beginning or in the middle of a sentence, it is verified whether the preceding token contains "B-material" or not. If a tag other than "B-material" is found, the check of the subsequent token is abandoned, and the token marked with "I-material" is not saved as a material class. A similar check is executed for the process class.

Once the material and process classes for a sentence are identified, the sentence, along with the corresponding material class or process class containing word/words, is saved in a database with the appropriate class labels and file name and doi of the paper. A MySql database system is used for storing this data, with a primary key automatically incremented for each row of data. Consequently, each data row contains an id, filename, doi, sentence, word/words, word class, or entity type (material or process).

2.5. Material-value and process-value relationship identification (phase 4)

The identification of material and relevant value relations, as well as the process with relevant value relations within a sentence, is based on rules. Four rules for the relationship between entity and value are established by examining the textual content of scientific documents in the EDLC domain. In this context, the entity refers to the word or words labeled material or process that were stored along with the sentence and other relevant information in the material and process identification step. The value refers to any number in floating point or integer format. The four rules are outlined below:

- Single entity - Single value
- Multiple entity - Single value
- Single entity - Multiple values

Table 4
Information arrangement of records in the database.

ID	*Paper / file name*	*Doi*	*Cleaned sentence*	*Lvalue*	*Word*	*Rvalue*	*Category*
1210	6.tei.xml.txt	10.1021/nn401818t	1 M NaOH is then poured with the mixture and set aside	1 M	NaOH	null null	Material

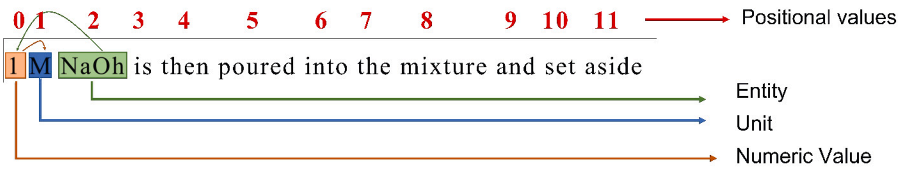

Fig. 5. Single entity - Single value operation.

- Multiple entity - Multiple values

As the sentences in the EDLC domain are stored with a single entity, regardless of whether the entity comprises a single word or multiple words, there is no need to match multiple entities in a single operation. For instance, the entity “sodium hydroxide”, consisting of two words and two tokens, namely B-mat and I-mat, is stored in the database with the entity type “material” and a single entity named “sodium hydroxide”. Hence, two rules addressing multiple entities are discarded, and the remaining two of the four established rules are considered. These rules are “Single Entity - Single Value” and “Single Entity - Multiple Values”, both dealing with the relationship between a single entity, which is a word or words labeled as material or process, and its corresponding value, which is any number in floating-point or integer format.

Prior to executing the rules, the presence of values in each record is checked. If a value exists, the position of the material or process class token is determined and saved for use in rule execution. In the EDLC domain of scientific papers, the value for a material or process entity is commonly mentioned in the same sentence in which the entity is mentioned and can appear either before or after the entity. Instances, where multiple material or process entities are mentioned with multiple values in the same sentence, are rare. In most cases, materials or processes are mentioned in separate sentences with corresponding values and units. Thus, for each entity, two value positions are defined, namely the left value and the right value. If no value is present on either side, it is marked as a null value, and the entity-value relationship is stored. It is also observed that the unit of a value appears as the adjacent word following the value. For instance, in the sentence “1 M NaOH is then poured with the mixture and set aside.”, “NaOH” is the material entity, and “1 M” is the value. To store this relationship, “1 M” is saved as the left value, “NaOH” as the token/word, “material” as the label/category, and “null” as the right value. The data arrangement is presented in Table 4.

2.5.1. Single entity - single value

This section outlines a rule to establish a relationship between a single entity (e.g., a material, process, or property) and a numerical value within a sentence. This rule applies to sentences that contain both the entity and the numerical value.

Initially, the position of the entity within the sentence is identified. Afterward, the position value of the numerical value needs to be determined. This involves locating where the numerical value is situated in the sentence. Subsequently, the position value of the numeric value is compared to the position value of the entity. The position value of the entity is the position in the sentence where the entity is located. If the position value of the numerical value is less than the position value of the entity, the numerical value is considered as the left value of the entity. This implies that the entity is situated to the right of the numerical value in the sentence. In contrast, if the position value of the numeric value is greater than the position value of the entity, the numerical value is designated as the right value of the entity. This signifies that the entity is located to the left of the numerical value in the sentence. This rule can be expressed by the following formula,

$$(\tau_p > v_p \implies v_{pl} \leftarrow v_p) \wedge (v_{pr} \leftarrow v_p)\{v_p, v_{pl}, v_{pr} \in \mathbb{R}\}$$

Here the position of the entity is τ_p, the position of the value is v_p, and the left and right values of the entity are v_{pl} and v_{pr}, respectively. Fig. 5 shows the process visually. Here the entity is marked with green, the numeric value with yellow, and the numeric value unit with blue. The position values for each token are marked in red.

This rule can also be expressed using the following regular expression,

```
(?P<word>\w+)\s*(?P<num>\d+) (?:\s+([a-zA-Z]+))?\b|
(?P<num>\d+) (?:\s+([a-zA-Z]+))?\s*(?P<word>\w+)\b
```

Here, `(?P<word>\w+)` captures a single word as a named group ‘word’. `\s*` matches any number of whitespace characters between the word and number. `(?P<num>\d+)\b` captures a single numerical value as a named group ‘num’ at the end of a word boundary. `|` is the OR operator. `(?P<num>\d+)\s*` captures a single numerical value as a named group ‘num’ at the start of the word boundary. `(?P<word>\w+)\b` captures a single word as a named group ‘word’ at the end of a word boundary. `(?:\s+([a-zA-Z]+))?` captures the word after the number, which is assumed to be the unit. The third capturing group is marked as NULL

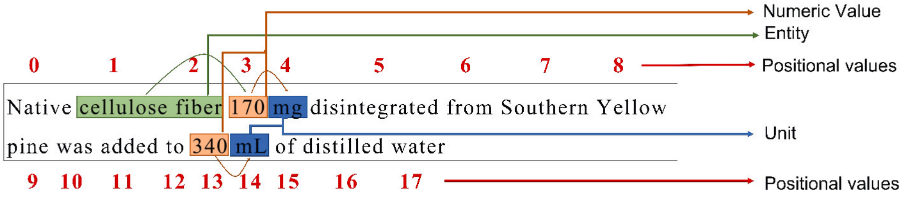

Fig. 6. Single entity - Multiple value operation.

if there is no word after the number. When the numerical value is found on one side, the other side would be marked as NULL automatically since it is not captured in the regular expression.

2.5.2. Single entity - multiple values

The single entity, multiple value rule applies to more than one value in a sentence. In this scenario, the closest value to the entity is considered. When looking at the records, it is noticeable that a related numerical value of an entity exists closer to that entity. Thus, if multiple numerical values occur in a record, the positions of the numerical values are measured, and the numerical value closest to the entity is assigned as the entity's value. If two numerical values contain the same position measure in absolute positional value, it is checked whether they are to the left or to the right of the entity. If left and right values are found, both numerical values are assigned as the entity's left and right numerical values and stored in the database. Fig. 6 shows the process visually. Here the entity was marked with green, the numeric value with yellow, and the numeric value unit with blue. The position values for each token are marked in red. This rule can also be expressed with the following formula,

$$v_p \forall \{v_{p1}, v_{p2}, \ldots, v_{pn}\}$$
$$\{$$
$$C_{pc} \leftarrow v_{p*}$$
$$C_{pp} \leftarrow |\tau_p - C_{pc}|$$
$$(C_{pp} > C_{pc} \implies C_{pp} \leftarrow C_{pc}(C_{pp} < \tau_p \implies v_{pl} \leftarrow C_{pp}) \wedge (v_{pr} \leftarrow C_{pp})) \wedge$$
$$(C_{pp} = C_{pc} \implies (C_{pp} > \tau_p \implies v_{pl} \leftarrow C_{pp}) \wedge (v_{pr} \leftarrow C_{pp}))\{v_p, v_{pl}, v_{pr} \in \mathbb{R}\}$$
$$\}$$

Here, C_{pc} is the current position of one out of multiple values, C_{pp} is the absolute positional distance from entity position and current value position, τ_p is the position of the entity, v_p is the position of the value, v_{pl} and v_{pr} are the left and right values of the entity respectively. Fig. 6 shows the process visually. Here the entity is marked with green, the numeric value with yellow, and the numeric value unit with blue. The position values for each token are marked in red.

This rule can also be expressed using the following regular expression,

```
\b(\w+)\b\s+(\d+(?:\.\d+)?)\s*(\w+)?\s+(\d+(?:\.\d+)?)\s*(\w+)?
\s*(\d+(?:\.\d+)?)\s*(\w+)?\b|\b(\d+(?:\.\d+)?)\s*(\w+)?
\s+(\w+)\b\s+(\d+(?:\.\d+)?)\s*(\w+)?\s*
(\d+(?:\.\d+)?)\s*(\w+)?\b
```

Here, `\b(\w+)\b` matches a word with word boundary on both sides and captures it. `\s+` matches one or more whitespace characters. `(\d+(?:\.\d+)?)` matches a numerical value and captures it. `\s*` matches zero or more whitespace characters. `(\w+)?` optionally matches a word and captures it. `|` is alternation to match either pattern and `\b` is word boundary on both sides to ensure the whole word is matched.

2.6. Knowledge graph generation (phase 5)

The knowledge graph is a representation of structured knowledge that is extracted, accumulated, or generated from various sources [50,51]. In this particular study, the knowledge of materials and processes is compiled from multiple scientific documents, making it a noteworthy effort to create a knowledge graph to store and organize this accumulated knowledge. This knowledge graph can be easily integrated into any system, enriching it with the information obtained from this research. Each knowledge graph can generate diverse types of knowledge through different queries. Knowledge graphs serve a vital role in organizing knowledge on the Internet, in data integration in enterprise systems, and as input-output in artificial intelligence systems.

A knowledge graph (KG) is a graph structure that establishes relationships between different entities. In this structure, the entities are represented as nodes, while the relationships between them are represented as directed edges with labeled meanings. Specifically, a KG is a subset of the cross product of $N \times L \times N$, where N is a set of nodes and L is a set of labels. Each member of this cross-product set is known as a triple and consists of a subject node, an object node, and a predicate or relationship label between them. The subset of the cross-product set can also be represented as $Subject \times Predicate \times Object$. This relationship is expressed in the following formula:

$$KG \subset N x L x N \equiv KG \subset Subject(node) \text{ x } Predicate(label) \text{ x } Object(node)$$

Fig. 7 shows the idea of a knowledge graph in the context of the current study. In this study, materials, processes, and values extracted from the scientific documents are called nodes of the knowledge graph, and the value relationship is called a label. In

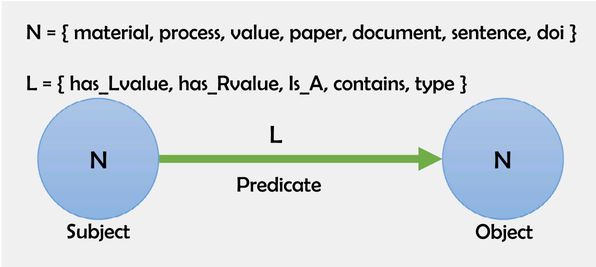

Fig. 7. Overview of knowledge graph structure.

addition to this triple, other triples were generated to integrate the knowledge graph with other systems. The triples generated for this knowledge graph are listed below.

1. Subject (material) x Predicate (has_Lvalue) x Object (value)
2. Subject (material) x Predicate (has_Rvalue) x Object (value)
3. Subject (process) x Predicate (has_Lvalue) x Object (value)
4. Subject (process) x Predicate (has_Rvalue) x Object (value)
5. Subject (paper_name) x Predicate (Is_A) x Object (document)
6. Subject (paper_name) x Predicate (contains) x Object (doi)
7. Subject (paper_name) x Predicate (contains) x Object (sentence)
8. Subject (paper_doi) x Predicate (contains) x Object (sentence)
9. Subject (sentence) x Predicate (contains) x Object (material)
10. Subject (sentence) x Predicate (contains) x Object (process)
11. Subject (material) x Predicate (type) x Object (material)
12. Subject (process) x Predicate (type) x Object (process)
13. Subject (value) x Predicate (type) x Object (value)
14. Subject (paper) x Predicate (type) x Object (paper)
15. Subject (paper) x Predicate (type) x Object (document)
16. Subject (doi) x Predicate (type) x Object (doi)
17. Subject (sentence) x Predicate (type) x Object (sentence)

The triples generated for the knowledge graph are optimized for the Resource Description Framework (RDF) [52] so they can be used with any system or software that can visualize or run queries on the knowledge graph. For example, Neo4j Graph Database [53] and GraphDB [54]. Each triple is stored in the N-triples format [55], a row-based plain text serialization format for storing RDF graphs. To generate the triples, the name of the paper, doi of the paper, sentence, token (process or material), and value (left and right value) are written to a simple N-triples file that uses the N-triples format and is stored as an nt file. Using this nt file, further visualizations and various SPARQL queries can be performed to analyze the knowledge and various patterns extracted from the scientific articles.

2.7. Evaluation metrics

The Named Entity Recognition (NER) model is evaluated with precision, recall, and $F1$ measures. Since our dataset is designed using the IOB segment boundary format, a single entity may consist of multiple tokens. For example, the entity "manganese oxide" contains two tokens, manganese and oxide. When evaluating entity predictions, the entity is marked as correct if each token is correctly labeled. True Positives (TP) are calculated when the entities are correctly predicted, and False Positives (FP) are marked when the predicted first token does not match the entity's marked token. False Negatives (FN) are marked when the system incorrectly predicts the first token of the predicted entities. The precision (P), recall (R), and the harmonic mean of precision and recall called $F1$, are computed using the equations (7), (8), and (9).

$$P = \frac{TP}{TP + FP} \tag{7}$$

$$R = \frac{TP}{TP + FN} \tag{8}$$

$$F1 = \frac{2 * P * R}{P + R} \tag{9}$$

The *Precision*, *Recall*, and $F1$ values are also used as performance measures for entity value extraction.

The link prediction task, which needs the embedding knowledge graph, was used to evaluate a knowledge graph. Knowledge graph embedding is discovering the semantic meaning of the knowledge graph's entities. Mean Rank (MR) and Mean Reciprocal Rank (MRR) are used to quantify link prediction performance. MR is the model's average rank position for all potential links,

Table 5
Document similarity score of MultipartiteRank keyword extraction algorithm for Jaccard, Cosine and Cosine with wordvector similarity indexes for positive and all sentences.

	Jaccard similarity index	*Cosine similarity index*	*Cosine with word vector similarity index*
Positive Sentence	0.14	0.25	0.92
All Sentence	0.14	0.25	0.91

which computes the arithmetic mean across all individual ranks (r), and MRR is the arithmetic mean of the reciprocal ranks, which is the inverse of the harmonic mean of the ranks (r). It is a metric that indicates the proportion of accurately anticipated triples. The equations (10) and (11) provide the values for MR and MRR, respectively.

$$\mathrm{MR} = \frac{1}{|\mathcal{I}|}\sum_{r\in\mathcal{I}} r \tag{10}$$

$$\mathrm{MRR} = \frac{1}{|\mathcal{I}|}\sum_{r\in\mathcal{I}} r^{-1} = \left(\frac{|\mathcal{I}|}{\sum_{r\in\mathcal{I}^{r-1}}}\right)^{-1} \tag{11}$$

Here, $\mathcal{I}$ is the set of all true triples rank and r is the rank position of the topmost ranked response.

2.8. Experimental setup

The Python programming language, version 3.6.5, in the Anaconda environment, is utilized to develop both the experiment codes and the proposed methodology. The Grobid version 0.7.0 is employed as a service with a Python wrapper for converting PDFs to plain text. The appropriate Python packages are employed for keyword extraction and sentence boundary identification, which are required for the document selection experiment. Tensorflow and Keras libraries for Python are used to develop, train, and predict machine learning models. The system hardware comprises an AMD Ryzen 4800H processor-based system with 24 GB RAM and a GTX 1650 4 GB GPU with CUDA environment. The system is operated on Windows 10. Ontotext GraphDB free edition is used to visualize the knowledge graph. The web-based prototype of the system is deployed through an Apache server, which supports the PHP scripting language.

3. Results and discussion

3.1. Appropriate paper selection

To select the appropriate document for the specified EDLC domain, the domain experts collect and review a list of 32 keywords and 10 documents containing these keywords relevant to the EDLC domain. These 10 documents were used as ground truth. The authors present the detailed experimental procedure in this paper [56]. The experiment shows that the MultipartiteRank keyword extraction technique and the cosine with wordvector similarity measure provide the best similarity score for the positive sentences extracted from the documents. The similarity score obtained by the unsupervised MultipartiteRank algorithm using the keywords provided by the experts is 92%. Based on the experimental result, the threshold value for the appropriate document selection procedure is set to 90% similarity. Only appropriate documents are investigated in the proposed system for material and process discovery, entity-value relationship identification, and knowledge graph generation. Table 5 shows the experimental result for selecting suitable documents. This table contains only the results of the best technique among all the techniques used for keyword extraction, namely MultipartiteRank. This algorithm produces the same similarity score for positive and all sentences set in Jaccard and Cosine similarity indices and different scores for positive and all sentences set in Cosine with the Wordvector similarity index. There is a very small difference of 1% between the two sets. The set with positive sentences achieved 92% similarity, while the set with all sentences achieved 91% similarity score. This is because the negative sentences used in the experiment contain very few or no keywords, as defined by the domain experts. The experimental results indicate that the dataset of positive sentences has a negligible impact of only 1% on the similarity score performance of the technique. The comparative similarity values for all the techniques used for the keyword extraction are shown in Fig. 8.

The proposed hybrid technique utilizing the MultipartiteRank algorithm and Cosine with wordvector significantly outperforms other keyword extraction strategies. MultipartiteRank is a better option to handle polysemous words and extract multi-word phrases as keywords because it considers the context in which they are used. Other techniques lack these capabilities. Moreover, the Cosine similarity with wordvector is able to capture important semantic and syntactic relationships between words, even when they are used in different contexts and have different meanings. The Jaccard similarity technique, which compares the presence or absence of elements, is not as effective as it does not take into account the weight or importance of the elements. The Cosine similarity is sensitive to the magnitude of the vectors, which can lead to higher scores for vectors that are dissimilar to the vectors of interest. The experimental results show that the proposed hybrid similarity technique outperforms the other techniques significantly.

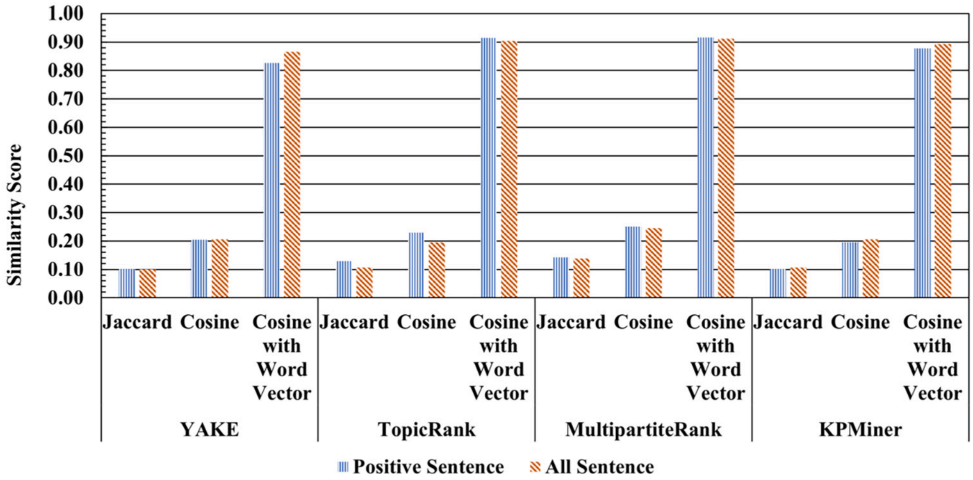

Fig. 8. Similarity scores for all the employed keyword extraction techniques found from the experimental result.

Table 6
Precision, Recall and F1 scores for different models employed in this work for the curated EDLC dataset.

Model name	*Precision*	*Recall*	*F1*
Deep Neural Network (DNN) model-MatRec	0.965	0.957	0.961
BERT-base-uncased	0.954	0.90	0.926
sciBERT-scivocav-uncased	0.946	0.93	0.938
Spacy-En_core_web_sm	0.642	0.198	0.303

3.2. Material and process identification

Identifying materials and processes in the scientific literature is a task of Named Entity Recognition, which also bears some resemblance to the task of sequence tagging due to the structure of the dataset created based on the annotated literature corpus of the EDLC domain. Therefore, a deep neural network model that uses Bi-LSTM layers among other layers is used to identify the materials and processes from the text. The model is trained and evaluated on a dataset created from 50 relevant scientific documents. In addition to training and evaluating the dataset with the proposed deep learning model, it is also trained and evaluated with some state-of-the-art models, namely BERT [57], sciBERT [58] and Spacy [23]. For the BERT model, the “Bert-base-uncased” variant is used, for the sciBERT model, the “sciBERT-scivocav-uncased” variant is used, and for the Spacy model, the “Spacy-En_core_web_sm” variant is used. The experimental result is shown in Table 6.

The reported Precision, Recall, and F1 values for each model in Table 6 are average values obtained through 5-fold cross-validation. The experimental results reveal that the proposed deep learning model outperforms other models in terms of F1 score (96%) for EDLC domain entities, which is higher than BERT (92%), sciBERT (93%), and Spacy (30%). Although all models were fine-tuned using the dataset curated in this study, the proposed model achieved better performance due to domain-specific training data and its size. Numerous studies have identified that LSTM-based models perform better than BERT or large language models when the LSTM model is trained with domain-specific data and relatively small training data [59,60]. BERT has been trained on large amounts of text data from various domains, including news articles, books, web pages, and Wikipedia. The pre-training is done on the general corpus of text data and is not specific to any particular domain [57] while SciBERT has been specifically trained on scientific text data. However, both models do not contain training data from the materials science domain, including papers from arXiv and PubMed. The pre-training was done using a large corpus of scientific text, including abstracts and full-text papers from the fields of computer science, physics, and biomedical science [58]. On the other hand, the proposed LSTM-based DNN model has been trained specifically with materials science domain data, taking into account the unique characteristics of the domain, leading to better performance than BERT and sciBERT. The Spacy model is based on CNN for the named entity recognition task, which is tuned for generic named entity recognition tasks like a person, geo-location, organizations, and others, and also lacks domain-specific training data, resulting in poor performance compared to the proposed model. The same trend was observed when all models were tested with another materials science dataset by Mysore et al. [61]. From Table 7, it can be seen that the proposed DNN model achieved a higher F1 score (96%) than BERT (93%) and sciBERT (94%). The changes observed in the scores of evaluation metrics are due to the increasing size of the training data. The experiment also provides a hint that the performance of the model is dependent on the type of task and data, and larger and more complex models may not always perform better in all types of tasks and data. In section 3.2.1, a statistical comparison of the F1 scores obtained by different models is discussed with respect to the z-test. In Fig. 9, the performance of the different models in terms of F1 score is shown. All the models used in this work have the same number of 36 epochs. For the proposed DNN model and the BERT model, the batch size is 16. For the sciBERT and the Spacy models, the batch

Table 7
Precision, Recall and F1 scores for different models employed in this work using the dataset by Mysore et al. [61].

Model name	*Precision*	*Recall*	*F1*
Deep Neural Network (DNN) model-MatRec	0.973	0.961	0.966
BERT-base-uncased	0.957	0.912	0.933
sciBERT-scivocav-uncased	0.948	0.936	0.941
Spacy-En_core_web_sm	0.683	0.335	0.449

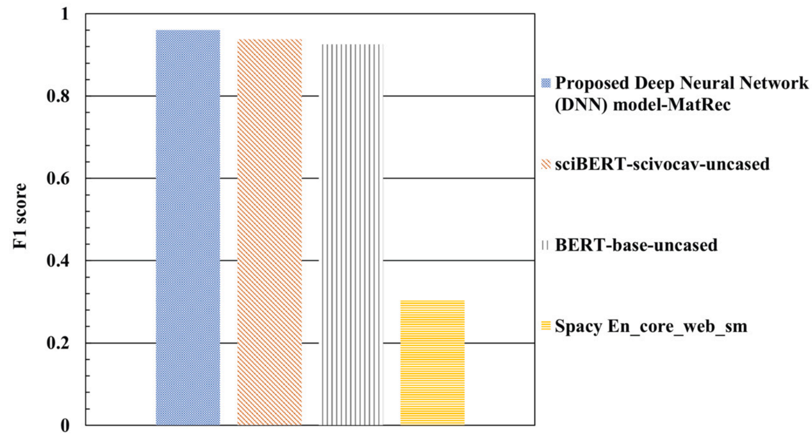

Fig. 9. Entity identification performance of different models in terms of F1 score.

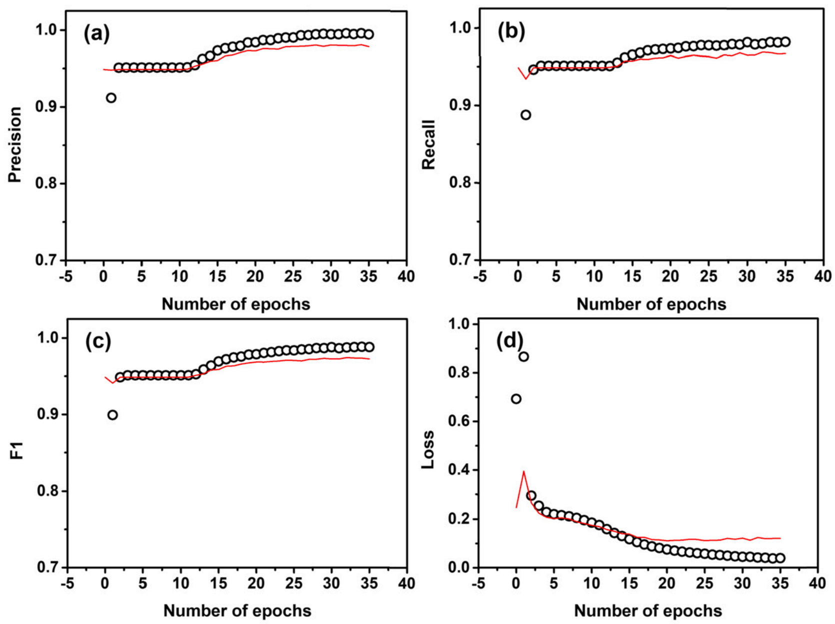

Fig. 10. Precision (a), Recall (b), F1 (c) and Loss (d) curve for the proposed deep neural network model for material and process identification from text. The solid lines represent the validation scores, and the circles represent the training scores for all the measures.

size is 8 due to the memory limitations of the GPU. All models are trained in a GPU environment. Details of the deep neural network model can be found in section 2.4 and the detailed experimental setup can be found in section 2.8 of the Supplementary Information.

Model training and validation insights of the proposed deep learning model can be seen in Fig. 10, which includes the precision curve, the recall curve, the F1 curve, and the loss curve, for the training and validation phases. Fig. 10a shows the precision curve for the training and validation phases of the model. The final precision values for the training and validation phases are 0.99 and 0.97, respectively. Fig. 10b and Fig. 10c show the recall and F1 curves for both the training and validation phases. The final recall values for the training and validation phases are 0.98 and 0.96, and the F1 values for the training and validation phases are 0.98 and 0.97, respectively. The final values for the training and validation losses are 0.03 and 0.11, respectively. The total training and validation losses are shown in Fig. 10d.

3.2.1. Statistical test (z-test)

In this study, we compared the performance of three models on a classification task. To determine whether the differences in their performance are statistically significant, we used a z-test. The z-test is a statistical test that compares two population means to determine whether they are different. In the context of machine learning, the z-test can be used to compare the performance of two or more models on a given task. By using a statistical test to evaluate the significance of the differences between the models, we can ensure that our results are reliable and not due to chance.

To calculate the z-test, we first need to calculate the standard error of the difference between the two models. We can use the formula:

$$SE = \sqrt{(p1 * (1 - p1)/n1) + (p2 * (1 - p2)/n2)}$$

where p1 and p2 are the proportions (F1 scores) of the two models, and n1 and n2 are the sample sizes (hold-out data sizes). We can then calculate the z-score using the formula:

$$z = (p1 - p2)/SE$$

where p1 and p2 are the proportions (F1 scores) of the two models.

Proposed model vs. BERT:

Model A (proposed model): F1 = 0.961, n1 = 1543

Model B (BERT): F1 = 0.926, n2 = 1543

$$SE = \sqrt{(0.961 * (1 - 0.961)/1543) + (0.926 * (1 - 0.926)/1543)}$$

$$SE = 0.008288503309$$

$$z = (0.961 - 0.926)/0.008288503309$$

$$z = 4.222716538$$

The z-score for this comparison is 4.222716538.

Proposed model vs. sciBERT:

Model A (proposed model): F1 = 0.961, n1 = 1543

Model C (sciBERT): F1 = 0.938, n2 = 1543

$$SE = \sqrt{(0.961 * (1 - 0.961)/1543) + (0.938 * (1 - 0.938)/1543)}$$

$$SE = 0.007872732008$$

$$z = (0.961 - 0.938)/0.007872732008$$

$$z = 2.921476303$$

The z-score for this comparison is 2.921476303.

In the z-test comparing Model A to Model B, the calculated z-value is 4.222716538, greater than the critical value of 1.96 at a significance level of 0.05. This indicates that Model A's F1 score significantly differs from Model B's F1 score, with a very high confidence level. This suggests that model A performs significantly better than model B regarding F1 score.

In the z-test comparing Model A to Model C, the calculated z-value is 2.921476303, also greater than the critical value of 1.96. This indicates that Model A's F1 score is significantly different from Model B's, with a high confidence level. This suggests that model A performs significantly better than model B regarding the F1 score. In the z-test comparing Model A to Model C, the calculated z-value is 2.921476303, also greater than the critical value of 1.96. This indicates that model A's F1 score significantly differs from model C's F1 score. This suggests that model A performs better than model C in terms of F1 score, but the difference is less significant than the difference between models A and B.

In both cases, the z-test determined the statistical significance of the differences in F1 scores between the two compared models. The results suggest that model A outperforms both model B and model C in terms of F1 score and ensures the reliability of the result of the experiment.

3.3. Entity-value relationship identification

Following the identification of materials and processes from the text follows the identification of the relationship with values to their respective materials or processes. To measure the performance of the entity-value relationship, a ground truth document is created that contains manually checked entity-value relationship data for 10 files selected by the domain experts. This ground truth file contains 366 records with a material-value relationship and 30 records with a process-value relationship. The experimental result of entity-value relationship identification is shown in Table 8.

Table 8
Precision, Recall and F1 score of Material and Process entities for entity-value relationship identification task.

Entity	*Precision*	Recall	F1
Material	0.705	1	0.827
Process	0.764	1	0.866

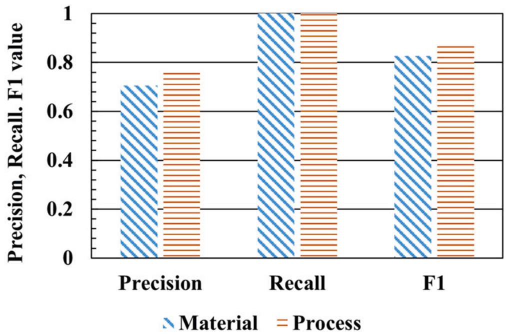

Fig. 11. Precision, Recall and F1 scores for Material and Process entities for entity-value relationship identification task.

From Table 8, it can be seen that the precision value for material and process entities is 70% and 76%, respectively, i.e., out of 100-time points, entities are correctly identified 70 and 76 times, respectively. This value is considerable at this stage because, in this scenario, the false positive rate is better than the false negative rate because this system helps the researchers to identify the relationships instead of calculating everything automatically in a production system. And this observation is also confirmed by the recall value for both entities, which is 100%. The F1 values for material and process entities are 82% and 86%, respectively. This is quite remarkable compared to the other works in this field. In Fig. 11, the precision, recall, and F1 values for both material and process entities are shown for identifying the entity-value relationship.

From the experiment, a recall value of 1 was achieved by the rule-based approach due to the availability of complete and accurate information in the ground truth data, which was used to design the approach. The ground truth document was created by domain experts and contains manually checked entity-value relationship data from 10 articles, which is considered a standard for measuring the performance of the entity-value relationship. It contains a total of 366 records with material-value relationships and 30 records with process-value relationships. The rule-based approach was specifically designed to match exactly with the information present in the ground truth data and was able to extract all 396 records. Therefore, a recall value of 1 was achieved, which indicates that the approach correctly identified all the material-value and process-value relationships present in the data. However, it is important to note that achieving a recall value of 1 in real-world scenarios can be challenging due to the inherent noise and variability of the data and which is not present in the experiment's ground truth data, and further research is needed to improve the approach's robustness and generalizability.

3.4. Knowledge graph

A knowledge graph is created using the process described in section 2.6 of the Supplementary Information using 17 rules. After the knowledge graph is generated, the relations are stored in N-triples format. To execute and test the generated knowledge graph, the file is loaded into GraphDB Free Edition. Using this software, the structure, relationships between subjects & objects, visualization of entities with relationships, and various knowledge queries are performed on the loaded knowledge graph. There are seven classes in the generated knowledge graph. The class Sentence contains the nodes labeled as Sentence, class doi contains the nodes labeled as doi numbers of documents, the class Paper contains the entries of scientific documents, the class Token contains the nodes labeled as Entity, class Label contains the entries of token labels, class Value contains the entries of value types, and class Document contains the nodes labeled as Document.

Fig. 12 shows a portion of the knowledge graph generated using data extracted from various scientific articles in the EDLC domain. The generated knowledge graph can be integrated into any Linked Open Data (LOD) cloud [62,63] considering that, the nodes convey the same meaning between different systems in the cloud. The generated knowledge graph can not only visualize the relationships between entities but also explore different patterns and connections between the classes and the members of the classes. For example, from this knowledge graph, it is easy to infer which doi numbers share the same materials or processes, which values for the same material are found in different scientific documents, and so on. In this way, different patterns and relationships between classes can be explored through this knowledge graph. Moreover, knowledge graphs provide the interface for machine-readable, interlinked data that can be shared among different intelligent systems through the semantic web with minimal effort [64].

To evaluate the completeness of the generated knowledge graph, a link prediction task is executed on the embedded knowledge graph using the Ampligraph [65] library. Three popular embedding models are used to embed the generated knowledge graph, namely, the ComplEx bilinear model [66], the HolE non-bilinear model [67], and the TransE pure translation model [68]. The

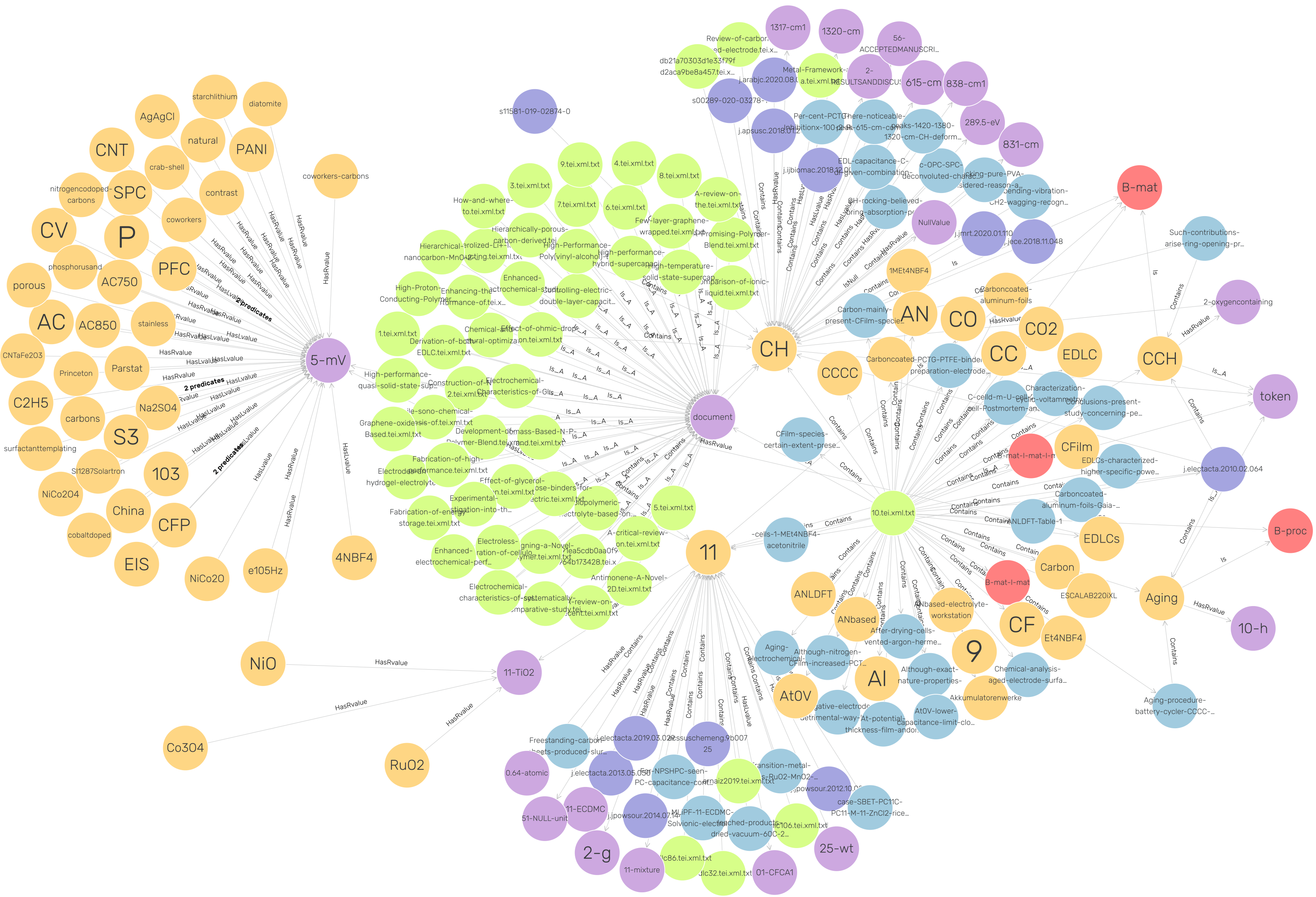

Fig. 12. Knowledge graph generated with different data extracted from different scientific papers relevant to EDLC domain.

Table 9
Mean Rank and Mean Reciprocal Rank scores obtained from the experiment for different models.

Model	*Generated KG - MRR*	*Generated KG - MR*
ComplEx	0.717	641
HolE	0.731	631
TransE	0.588	382

experimental results are shown in Table 9. MR and MRR are two of the knowledge graph completion evaluation indicators, and the results are shown with regard to these terms for all models and datasets.

The results show that the knowledge graph generated from the extracted information performs differently in different models. The ComplEx model has the highest MR value of 641, the HolE model has a slightly lower MR value of 631, and the TransE model has the lowest MR value of 382. This indicates that the ComplEx model is more effective in capturing the complexity and diversity of the extracted information. Regarding MRR, the HolE model performs slightly better than the ComplEx model, with an MRR of .731 compared to .717. The TransE model has the lowest MRR of .538, which indicates that it is not as effective in capturing the relationships between different entities in the knowledge graph. These scores from the evaluation metrics indicate that the knowledge graph is well-structured and can be used to retrieve information efficiently.

The knowledge graph has significant contributions to the field of materials science as it can be used to extract and organize information related to materials science research efficiently. It can help researchers to identify and retrieve relevant information, which can lead to the discovery of new materials, improved materials, and advanced materials applications. The knowledge graph can also help to identify knowledge gaps and areas of research that require further investigation. In summary, the knowledge graph is a valuable tool for materials science researchers and can potentially contribute to significant advancements in the field.

3.5. Discussion

This research focuses on extracting materials and processes used in the EDLC domain, where all the materials are not chemicals, organic, or inorganic. The materials used in this domain are a mixture of organic and inorganic chemicals. The data and systems provided in the studies [69,10–12] are based on static chemical entities and chemical formulas. At the same time, this study provides a method to extract not only chemical data but also a wide range of materials and processes by training a small number of scientific documents that are not static in terms of data discovery. Once trained, the system can extract entities from text that exist in or outside of a static database.

Besides these studies, some studies in the field of materials science have also been carried out by [19–21,15,13]. All the studies that focus on the field of materials science have one thing in common: they focus on specific materials or only on the synthesis methods. In the work of Guha et al. [20] they extracted five different types of information, such as materials, processes, parameters, formulas, and structures, from the scientific literature in materials science. They are only interested in specific materials and related information mentioned in the title or system user input. On the other hand, this proposed work is specifically concerned with materials and processes in the EDLC domain. There is no option for human intervention or title extraction as the documents are marked as relevant or irrelevant to the domain by the appropriate paper selection mechanism of the proposed system before extracting the entities from the scientific literature. Moreover, the proposed tool can extract associated values for the entities and represent the entire knowledge in a graph-based knowledge map. The proposed tool also uses the Bi-LSTM layer within the deep neural network model like the studies mentioned in [19,20,13]. Among all the works mentioned, the proposed one received the highest F1 score of 96% whereas [19] achieved 87%, and [20] achieved 73.13% F1 score. The work presented in [13] deals with inorganic materials, a fixed synthesis process, and static conditions for heating and mixing. At the same time, the proposed system considers all possible materials, processes, and values relevant to the materials and processes. In this work, data collection from PDF documents was also discarded, whereas the proposed system uses PDF documents as most of the scientific documents are available in this format.

Most related works used abstracts from the papers to train the system for named entity recognition and used paragraphs to identify relevant entities, except for the work mentioned in [20]. This study works with sentences and marks the sentences as relevant or non-relevant to the domain based on the presence of entities in the sentences. This means that the NER task should also be performed for non-relevant sentences. On the other hand, the proposed system also works with sentence-level training and extraction. And this system selects the matching document not based on the sentences but on keywords provided by the domain expert. This reduces the task of checking each document sentence to find an entity and marking it as relevant. Relevance in this tool is measured at the document level, not the sentence level. Apart from this, the proposed system was trained using only 6250 sentences, while the system provided by [20] uses 49610 sentences and provides a lower F1 score than the proposed system. All the related studies using deep learning models [20,19,15,13], which use Word2Vec and ELMO embedding in addition to character embedding, have a lower F1 score than the proposed system which uses a generic embedding layer in the neural network model.

Few works mention relevant document processing before extracting information from documents. [20] mentioned sentence-level relevance, and [13] mentioned paragraph-level relevance. For both sentence-level and paragraph-level relevance, the whole document must be processed first, and then the sentences or paragraphs should be extracted and checked for relevance. Both approaches require different machine learning classification models to label the relevance of sentences or paragraphs. On the other hand, the proposed system uses keyword-based relevance measurement to determine how important a document is before it pulls information from it. This means there is no need to create a new dataset and train a new model to determine how relevant a document is.

The proposed deep neural network model performed better on the NER task than the pre-trained models SciBERT [58] and BERT [57], despite the fact that both pre-trained models were fine-tuned using the same dataset.

The most challenging part of this study is to extract the entity-value relationship from the literature, and the proposed system has a considerable F1 score with respect to this task, which is 82% and 86% for material and process entities, respectively. In addition to named entity recognition and entity value extraction, this paper also presented a knowledge graph representation of the knowledge extracted from the scientific literature. Knowledge graphs are useful for finding hidden patterns and relationships between different entities. Moreover, the whole knowledge can be shared or integrated with any system that uses common entities. For example, the knowledge graph made for the EDLC domain can be easily added to an existing chemical database or system for materials science. This makes collaborating and sharing knowledge easier, which is impossible in any related work discussed in this paper. The next section discusses the potential applications of the proposed tool.

3.5.1. Potential applications of the tool

The tool described in this paper has potential applications in several areas. One such area is scientific research, where the tool can automatically retrieve relevant scientific articles and extract the necessary information for data analysis, saving researchers a considerable amount of time and effort. Additionally, the tool can be used in industries that require large amounts of data analysis, such as the materials science industry. The ability to accurately extract material science entities and their associated numerical values and create knowledge graphs from this information can significantly improve the efficiency and accuracy of data analysis. The tool can also be used in educational settings, where it can assist students and educators in finding relevant scientific articles and analyzing them.

The tool can help researchers and engineers quickly find relevant papers and extract key information such as material properties and processing conditions in the manufacturing industry. This can accelerate the development of new materials and improve the quality and efficiency of manufacturing processes. In research and development, the tool can aid researchers in identifying key materials and processing parameters from the scientific literature. This can help researchers to better understand the behavior of materials and improve the design of new materials with desirable properties.

Moreover, with the advancement of the fourth industrial revolution, where artificial intelligence and machine learning play a significant role in transforming various industries, the proposed tool can be used as a building block for developing intelligent systems in materials science. The tool can also be used in developing knowledge-based systems and chatbots to assist engineers and researchers in the field.

Overall, the proposed tool has the potential to significantly improve the efficiency and accuracy of information extraction from scientific literature and can be used to accelerate research and development in the materials science field.

3.6. Online interface and data availability

Of the 250 scientific documents collected, 50 are annotated to create the material and process identification task dataset. The remaining 200 documents are executed in the system. Out of the 200 papers, the system has processed 160 papers, and 40 papers have been found unsuitable during the appropriate paper selection phase. Using the DNN part of the system, material and process entities from 160 papers are identified along with the relevant values of the entities and a knowledge graph is created. All this processed data is entered into an information system accessible at http://matrec.aiubcc.org. The knowledge graph's N-triples file is also available in the mentioned web application under the "Knowledge Graphs" section. The code and data can be obtained from this [https://github.com/ping543f/matrec-paper] GitHub repository by opening an issue or emailing the corresponding author. The sample entity recognition dataset [70] can be downloaded from Mendeley Data at the url https://data.mendeley.com/datasets/s3st6n77pr.

4. Conclusion

This study proposes a Deep Learning based material information system for electric double layer capacitor. This material information system is capable of acquiring and processing data. It is also capable of suggesting relevant articles for the EDLC domain. A new hybrid approach consisting of a keyword extraction and document similarity techniques is used to find the relevant documents. Materials, processes, and their associated values are extracted from the relevant articles using a deep learning model for entity extraction and a rule-based approach for value extraction. To train the deep learning model for entity extraction, 50 documents were annotated with the material and process class. The entity extraction model achieved an F1 score of 96%, and the entity value relationship extraction model achieved an F1 score of 82% and 86% for material and process entities, respectively. The model was applied to 200 articles from 2015 to 2020 relevant to EDLC. The information extracted from these articles is hosted in a publicly available web application. In addition, a knowledge graph is created from the extracted knowledge, and the downloadable N-triples file for the knowledge graph is also provided in the web application.

The proposed system identifies the relevant documents for the specified domain using a hybrid approach of keyword extraction and document similarity indices. This approach can be extended to any generic domain. The entity-value relationship is identified using a rule-based approach, and this identification process can be extended using a graph-based approach and dependency trees. The system can also be tested with big data infrastructure called "material big data informatics". Moreover, our approach can also be extended to other named entity recognition tasks beyond the material and process domains. The underlying architecture of our model can be used for other tasks, such as identifying named entities in the medical, legal, and financial domains, with suitable modifications to the training data. Our approach can be a useful starting point for other researchers working on named entity recognition tasks in different domains.

Funding statement

This work was supported by the Directorate of Research and Community Service, Telkom University. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

CRediT authorship contribution statement

M. Saef Ullah Miah: Conceived and designed the experiments; Performed the experiments; Analyzed and interpreted the data; Wrote the paper.

Junaida Sulaiman; Rajan Jose: Conceived and designed the experiments; Analyzed and interpreted the data; Wrote the paper.

Talha Bin Sarwar: Performed the experiments; Analyzed and interpreted the data.

Nur Ibrahim; Md Masuduzzaman: Contributed reagents, materials, analysis tools or data; Wrote the paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data associated with this study has been deposited at https://data.mendeley.com/datasets/s3st6n77pr.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.heliyon.2023.e20003.

References

[1] Google, Google scholar, https://scholar.google.com/, 2021.
[2] S. Ramakrishna, R. Jose, Addressing sustainability gaps, Sci. Total Environ. 806 (2022) 151208, https://doi.org/10.1016/j.scitotenv.2021.151208.
[3] Nippon Chemicon, Introduction of Electric Double Layer Capacitor, Tech. Rep., Nippon Chemi-Con Corporation, 2020, https://chemi-con.com/wp-content/uploads/2021/05/DXF-Series.pdf.
[4] P.J. Hall, M. Mirzaeian, S.I. Fletcher, F.B. Sillars, A.J. Rennie, G.O. Shitta-Bey, G. Wilson, A. Cruden, R. Carter, Energy storage in electrochemical capacitors: designing functional materials to improve performance, Energy Environ. Sci. 3 (9) (2010) 1238–1251, https://doi.org/10.1039/c0ee00004c.
[5] B. Pal, A. Yasin, R. Kaur, M. Tebyetekerwa, F. Zabihi, S. Yang, C.C. Yang, Z. Sofer, R. Jose, Understanding electrochemical capacitors with in-situ techniques, Renew. Sustain. Energy Rev. 149 (2021) 111418, https://doi.org/10.1016/J.RSER.2021.111418.
[6] Y. Wang, Q. Qu, S. Gao, G. Tang, K. Liu, S. He, C. Huang, Biomass derived carbon as binder-free electrode materials for supercapacitors, https://doi.org/10.1016/j.carbon.2019.09.018, Dec. 2019.
[7] Y. Gao, L. Zhao, Review on recent advances in nanostructured transition-metal-sulfide-based electrode materials for cathode materials of asymmetric supercapacitors, Chem. Eng. J. 430 (2022) 132745, https://doi.org/10.1016/J.CEJ.2021.132745.
[8] G. Jiang, R.A. Senthil, Y. Sun, T.R. Kumar, J. Pan, Recent progress on porous carbon and its derivatives from plants as advanced electrode materials for supercapacitors, J. Power Sources 520 (2022) 230886, https://doi.org/10.1016/J.JPOWSOUR.2021.230886.
[9] P. Forouzandeh, P. Ganguly, R. Dahiya, S.C. Pillai, Supercapacitor electrode fabrication through chemical and physical routes, J. Power Sources 519 (2022) 230744, https://doi.org/10.1016/J.JPOWSOUR.2021.230744.
[10] M.C. Swain, J.M. Cole, ChemDataExtractor: a toolkit for automated extraction of chemical information from the scientific literature, J. Chem. Inf. Model. 56 (10) (2016) 1894–1904, https://doi.org/10.1021/acs.jcim.6b00207.
[11] T. Rocktäschel, M. Weidlich, U. Leser, ChemSpot: a hybrid system for chemical named entity recognition, Bioinformatics 28 (12) (2012) 1633–1640, https://doi.org/10.1093/BIOINFORMATICS/BTS183, https://academic.oup.com/bioinformatics/article/28/12/1633/266861.
[12] D.M. Jessop, S.E. Adams, E.L. Willighagen, L. Hawizy, P. Murray-Rust, OSCAR4: a flexible architecture for chemical textmining, J. Cheminform. 3 (10) (2011) 1–12, https://doi.org/10.1186/1758-2946-3-41/TABLES/2, https://jcheminf.biomedcentral.com/articles/10.1186/1758-2946-3-41.
[13] O. Kononova, H. Huo, T. He, Z. Rong, T. Botari, W. Sun, V. Tshitoyan, G. Ceder, Text-mined dataset of inorganic materials synthesis recipes, Sci. Data 6 (1) (2019) 203, https://doi.org/10.1038/s41597-019-0224-1.
[14] A. Friedrich, H. Adel, F. Tomazic, J. Hingerl, R. Benteau, A. Marusczyk, L. Lange, The SOFC-exp corpus and neural approaches to information extraction in the materials science domain, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2020, pp. 1255–1268, https://aclanthology.org/2020.acl-main.116, Online.
[15] S. Mysore, E. Kim, E. Strubell, A. Liu, H.-S. Chang, S. Kompella, K. Huang, A. McCallum, E. Olivetti, Automatically extracting action graphs from materials science synthesis procedures, arXiv:1711.06872, 2017.
[16] E.A. Olivetti, J.M. Cole, E. Kim, O. Kononova, G. Ceder, T.Y.J. Han, A.M. Hiszpanski, Data-driven materials research enabled by natural language processing and information extraction, Appl. Phys. Rev. 7 (4) (2020), https://doi.org/10.1063/5.0021106.
[17] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, M. Hoffmann, Industry 4.0, Bus. Inf. Syst. Eng. 6 (4) (2014) 239–242.
[18] R. Jose, S. Ramakrishna, Materials 4.0: materials big data enabled materials discovery, Appl. Mater. Today 10 (2018) 127–132.
[19] L. Weston, V. Tshitoyan, J. Dagdelen, O. Kononova, A. Trewartha, K.A. Persson, G. Ceder, A. Jain, Named entity recognition and normalization applied to large-scale information extraction from the materials science literature, J. Chem. Inf. Model. 59 (9) (2019) 3692–3702, https://doi.org/10.1021/acs.jcim.9b00470.

[20] S. Guha, A. Mullick, J. Agrawal, S. Ram, S. Ghui, S.-C. Lee, S. Bhattacharjee, P. Goyal, Matscie: an automated tool for the generation of databases of methods and parameters used in the computational materials science literature, Comput. Mater. Sci. 192 (2021) 110325.
[21] V. Tshitoyan, J. Dagdelen, L. Weston, A. Dunn, Z. Rong, O. Kononova, K.A. Persson, G. Ceder, A. Jain, Unsupervised word embeddings capture latent knowledge from materials science literature, Nature 571 (7763) (2019) 95–98, https://doi.org/10.1038/s41586-019-1335-8.
[22] L. Patrice, F. Luca, GROBID, https://github.com/kermitt2/grobid, 2008–2021, GitHub, swh:1:dir:dab86b296e3c3216e2241968f0d63b68e8209d3c.
[23] M. Honnibal, M. Johnson, An improved non-monotonic transition system for dependency parsing, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 1373–1378, https://aclweb.org/anthology/D/D15/D15-1162.
[24] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, C. Dyer, Neural architectures for named entity recognition, arXiv preprint, arXiv:1603.01360.
[25] Z. Huang, W. Xu, K. Yu, Bidirectional lstm-crf models for sequence tagging, arXiv preprint, arXiv:1508.01991.
[26] M.S.U. Miah, J. Sulaiman, T.B. Sarwar, A. Naseer, F. Ashraf, K.Z. Zamli, R. Jose, Sentence boundary extraction from scientific literature of electric double layer capacitor domain: tools and techniques, Appl. Sci. 12 (3) (2022) 1352.
[27] UMP Library, e-Resource UMP Lib, https://login.ezproxy.ump.edu.my/login, 2021.
[28] L.A. Ramshaw, M.P. Marcus, Text chunking using transformation-based learning, in: Natural Language Processing Using Very Large Corpora, Springer, 1999, pp. 157–176.
[29] UBIAI Web Services, UBIAI easy to use text annotation tool, https://ubiai.tools/Docs, 2021.
[30] L.M. Hsu, R. Field, Interrater agreement measures: comments on kappan, Cohen's kappa, Scott's pi, and Aickin's alpha, Underst. Stat. 2 (3) (2003) 205–219, https://doi.org/10.1207/S15328031US0203_03.
[31] M.L. McHugh, Interrater reliability: the kappa statistic, Biochem. Med. 22 (3) (2012) 276–282.
[32] Grammarly, Negatives and negation–grammar rules | grammarly, https://www.grammarly.com/blog/negatives/, 2021.
[33] J. Col, Negative vocabulary word list - enchanted learning, https://www.enchantedlearning.com/wordlist/negativewords.shtml, 1998.
[34] Y. HaCohen-Kerner, H. Badash, Positive and Negative Sentiment Words in a Blog Corpus Written in Hebrew, Procedia Computer Science, vol. 96, Elsevier B.V., 2016, pp. 733–743.
[35] P. Jaccard, The distribution of the flora in the Alpine zone, New Phytol. 11 (2) (1912) 37–50, https://doi.org/10.1111/j.1469-8137.1912.tb05611.x.
[36] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Inf. Process. Manag. 24 (5) (1988) 513–523, https://doi.org/10.1016/0306-4573(88)90021-0.
[37] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint, arXiv:1301.3781.
[38] I. Witten, G. Paynter, E. Frank, C. Gutwin, C. Nevillmanning, kea: practical automatic keyphrase extraction, in: Proceedings of the Fourth ACM Conference on Digital, Libraries, 1999, pp. 254–255.
[39] T.D. Nguyen, M.-T. Luong, Wingnus: keyphrase extraction utilizing document logical structure, in: Proceedings of the 5th International Workshop on Semantic Evaluation, 2010, pp. 166–169.
[40] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, A. Jatowt, YAKE! Keyword extraction from single documents using multiple local features, Inf. Sci. 509 (2020) 257–289, https://doi.org/10.1016/J.INS.2019.09.013.
[41] A. Bougouin, F. Boudin, B. Daille, Topicrank: graph-based topic ranking for keyphrase extraction, in: International Joint Conference on Natural Language Processing (IJCNLP), 2013, pp. 543–551.
[42] F. Boudin, Unsupervised keyphrase extraction with multipartite graphs, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 667–672, https://www.aclweb.org/anthology/N18-2105.
[43] S.R. El-Beltagy, A. Rafea, Kp-miner: a keyphrase extraction system for English and Arabic documents, Inf. Sci. 34 (1) (2009) 132–144.
[44] G. Van Rossum, F.L. Drake, Python 3 Reference Manual, CreateSpace, Scotts Valley, CA, 2009.
[45] F. Chollet, et al., Keras, https://github.com/fchollet/keras, 2015.
[46] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: large-scale machine learning on heterogeneous systems, software available from https://www.tensorflow.org/, 2015.
[47] M. Honnibal, I. Montani, spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing, to appear, 7 (1) (2017) 411–420.
[48] S. Bird, E. Klein, E. Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit, O'Reilly Media, Inc., 2009.
[49] R. Rehurek, P. Sojka, Gensim–python framework for vector space modelling, vol. 3 (2), NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic, 2011.
[50] C.K. Baru, What is a knowledge graph?, https://web.stanford.edu/class/cs520/2020/notes/What_is_a_Knowledge_Graph.html, 2020.
[51] S. Yang, X. Cai, Bilateral knowledge graph enhanced online course recommendation, Inf. Sci. 107 (2022) 102000, https://doi.org/10.1016/j.is.2022.102000.
[52] V. Christophides, Resource Description Framework (RDF) Schema (RDFS), Springer US, Boston, MA, 2009, pp. 2425–2428, Ch. R.
[53] Neo4j, Neo4j documentation, https://neo4j.com/docs/, May 2021.
[54] Ontotext, Graphdb, https://graphdb.ontotext.com/documentation/standard/.
[55] D. Beckett, Rdf 1.1 n-triples, https://www.w3.org/TR/n-triples.
[56] M. Miah, J. Sulaiman, T.B. Sarwar, K.Z. Zamli, R. Jose, Study of keyword extraction techniques for electric double-layer capacitor domain using text similarity indexes: an experimental analysis, Complexity (2021).
[57] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186, https://aclanthology.org/N19-1423.
[58] I. Beltagy, K. Lo, A. Cohan, Scibert: a pretrained language model for scientific text, arXiv preprint, arXiv:1903.10676.
[59] K. Zeberga, M. Attique, B. Shah, F. Ali, Y.Z. Jembre, T.-S. Chung, A novel text mining approach for mental health prediction using bi-lstm and bert model, Comput. Intell. Neurosci. (2022).
[60] A. Ezen-Can, A comparison of lstm and bert for small corpus, https://doi.org/10.48550/ARXIV.2009.05451, https://arxiv.org/abs/2009.05451, 2020.
[61] S. Mysore, Z. Jensen, E. Kim, K. Huang, H.-S. Chang, E. Strubell, J. Flanigan, A. McCallum, E. Olivetti, The materials science procedural text corpus: annotating materials synthesis procedures with shallow semantic structures, in: Proceedings of the 13th Linguistic Annotation Workshop, Association for Computational Linguistics, Florence, Italy, 2019, pp. 56–64, https://www.aclweb.org/anthology/W19-4007.
[62] P. Pico-Valencia, J.A.H. Terrizam, P. Paderewski, A systematic method for building Internet of agents applications based on the linked open data approach, Future Gener. Comput. Syst. 94 (2019) 250–271.
[63] P. Basile, C. Greco, A. Suglia, G. Semeraro, Bridging the gap between linked open data-based recommender systems and distributed representations, Inf. Sci. 86 (2019) 1–8, https://doi.org/10.1016/j.is.2019.07.001.
[64] G. Hübscher, V. Geist, D. Auer, A. Ekelhart, R. Mayer, S. Nadschläger, J. Küng, Graph-based managing and mining of processes and data in the domain of intellectual property, Inf. Sci. 106 (2022) 101844, https://doi.org/10.1016/j.is.2021.101844.

[65] L. Costabello, S. Pai, C.L. Van, R. McGrath, N. McCarthy, P. Tabacof, AmpliGraph: a library for representation learning on knowledge graphs, https://doi.org/10.5281/zenodo.2595043, Mar. 2019.
[66] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: International Conference on Machine Learning, PMLR, 2016, pp. 2071–2080.
[67] M. Nickel, L. Rosasco, T. Poggio, Holographic embeddings of knowledge graphs, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30, 2016, pp. 1955–1961.
[68] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, Adv. Neural Inf. Process. Syst. 26 (2013).
[69] J. Li, Y. Sun, R.J. Johnson, D. Sciaky, C.H. Wei, R. Leaman, A.P. Davis, C.J. Mattingly, T.C. Wiegers, Z. Lu, BioCreative V CDR task corpus: a resource for chemical disease relation extraction, Database 2016 (2016) baw068, https://doi.org/10.1093/DATABASE/BAW068, https://academic.oup.com/database/article/doi/10.1093/database/baw068/2630414.
[70] M.S.U. Miah, J. Sulaiman, R. Jose, T.B. Sarwar, Matrecdata: material and process named entity recognition dataset for edlc, https://data.mendeley.com/datasets/s3st6n77pr, 2023, https://doi.org/10.17632/s3st6n77pr.2.