Applied Soft Computing 112 (2021) 107765

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A Multi-Channel Convolutional Neural Network approach to automate the citation screening process

Raymon van Dinter [a], Cagatay Catal [b,*], Bedir Tekinerdogan [a]

[a] Information Technology Group, Wageningen University & Research, Wageningen, The Netherlands
[b] Computer Science and Engineering, Qatar University, Doha, Qatar

ARTICLE INFO

Article history:
Received 18 March 2021
Received in revised form 26 July 2021
Accepted 27 July 2021
Available online 30 July 2021

Keywords:
Systematic literature review (SLR)
Citation screening
Automation
Neural networks
Natural language processing

ABSTRACT

The systematic literature review (SLR) process is separated into several steps to increase rigor and reproducibility. The selection of primary studies (i.e., citation screening) is an important step in the SLR process. The citation screening process aims to identify the relevant primary studies fairly and with high rigor using selection criteria. Through the study selection criteria, reviewers determine whether an article should be included or excluded from the SLR. However, the screening process is highly time-consuming and error-prone as the researchers must read each title and possibly hundreds to thousands of abstracts and full-text documents. This study aims to automate the citation screening process using Deep Learning algorithms. With this, it is aimed to reduce the time and costs of the citation screening process and increase the precision and recall of the relevant primary studies. A Multi-Channel Convolutional Neural Network (CNN) is proposed, which can automatically classify a given set of citations. As the architecture uses the title and abstract as features, our end-to-end pipeline is domain-independent. We have performed six experiments to assess the performance of Multi-Channel CNNs across 20 publicly available systematic literature review datasets. It was shown that for 18 out of 20 review datasets, the proposed method achieved significant workload savings of at least 10%, while in several cases, our model yielded a statistically significantly better performance over two benchmark review datasets. We conclude that Multi-Channel CNNs are effective for the citation screening process in SLRs. Multi-Channel CNNs perform best on large datasets of over 2500 samples with few abstracts missing.

1. Introduction

A systematic literature review is a means of identifying, evaluating, and synthesizing all available research relevant to a particular research question, or topic area, or phenomenon of interest [1]. It is well-known and often used in the medical domain, but it has also spread to other domains such as software engineering in recent years. The systematic literature review process is separated into several steps to increase rigor and reproducibility. The selection of primary studies (i.e., citation screening) is an important step of this process. The primary study selection process attempts to identify the relevant primary studies fairly and with high rigor using study selection criteria. Through the study selection criteria, reviewers determine whether an article is either included or excluded from the systematic literature review.

However, there are downsides to this process. First, an experienced reviewer is estimated to screen up to two articles per minute [2], and as a reviewer must read each title and possibly hundreds to thousands of abstracts and full-text documents, the screening process is highly time-consuming. With a median of eight months after the last search, a systematic review is often outdated before publication as they take so much time to produce [3]. Second, as reviewers must read so many articles, the review process is highly repetitive, which is sensitive to errors. This results in that reviewers find it challenging to include the most relevant articles (i.e., high recall) while excluding the most irrelevant articles (i.e., high precision).

As reviewers must determine an article's inclusion, the citation screening process can be viewed as a binary classification problem [4–7]. As a result, machine learning methods can reduce time costs and increase precision and recall. Although current automatic text classification methods have achieved substantial workload savings [5,8], they still use mainly shallow machine learning algorithms. Deep learning techniques have favorable characteristics, such as avoiding hand-crafted features, smooth application on new domains, and the use of end-to-end models that resolve classical shallow learning challenges. Furthermore, to our knowledge, text classification using deep learning has not yet been applied in the systematic literature review domain.

* Corresponding author.
E-mail addresses: raymon.vandinter@wur.nl (R. van Dinter), ccatal@qu.edu.qa (C. Catal), bedir.tekinerdogan@wur.nl (B. Tekinerdogan).

https://doi.org/10.1016/j.asoc.2021.107765

List of abbreviations

ACE Inhibitors	Angiotensin-converting enzyme (ACE) inhibitors
ADHD	Attention Deficit/Hyperactivity Disorder
API	Application programming interface
BERT	Bidirectional Encoder Representations from Transformers
BPA	Bisphenol A
CNN	Convolutional Neural Network
FN	False Negative
FTE	Full-Time Equivalent
GloVe	Global Vectors for Word Representation
GPT-3	Generative Pre-trained Transformer 3
GPU	Graphics processing unit
HIV	human immunodeficiency viruses
IDF	Inverse Document Frequency
IMDB	Internet Movie Database
IQR	Inter-Quartile Range
LSTM	Long Short-Term Memory
MANN	Memory-augmented Neural Network
MeSH	Medical Subject Headings
NB	Naïve Bayes
NLP	Natural Language Processing
NSAID	Non-Steroidal Anti-Inflammatory Drug
PFOA	Perfluorooctanoic acid
PFOS	Perfluorooctanesulfonic acid
RNN	Recurrent Neural Network
SLR	Systematic Literature Review
SVM	Support Vector Machine
SWIFT	Sciome Workbench for Interactive computer-Facilitated Text-mining
TF	Term Frequency
TFIDF	Term Frequency–Inverse Document Frequency
TN	True Negative
TP	True Positive
WSS	Work Saved over Sampling
WSS@95%	Work Saved over Sampling at a Recall of 95%

We propose a Multi-Channel Convolutional Neural Network (CNN) approach to support the automated classification of primary studies. The approach requires a set of manually labeled citations to learn to generalize between included and excluded samples. Consecutively, the trained model automatically identifies unlabeled citations, reducing the high human workload involved with the citation screening task. This Multi-Channel CNN approach can significantly reduce human workload in the systematic literature review process, as reviewers only need to label a subset of the literature. Furthermore, our approach can be used in other literature review methodologies, for instance, Cochrane Reviews, Rapid Reviews, Mapping Studies, and other NLP-related tasks such as sentiment analysis.

A recent SLR study on the automation of SLR studies by van Dinter, Tekinerdogan and Catal [9] shows that most existing semi-automatic citation screening methods adopt document representation techniques, such as bag-of-words and TF-IDF, that rely on words' frequency. Therefore, the feature representation of documents naturally ignores the readily available information on the context of those words. Furthermore, [9] show that most studies use domain-dependent document metadata, such as Medical Subject Headings (MeSH).

This paper presents a domain-independent Multi-Channel CNN approach that leverages the meaning of keywords and sentences from the title and abstract of training samples to generate informative document features. The proposed method uses parallel CNN architectures with varying kernel sizes followed by a feed-forward neural network to learn essential words and phrases for the citation screening process. More specifically, our proposed neural network uses the respected and widely used Glove Embeddings to gain insight from each word's context. The embedding matrices gained from each document can be integrated with any classification algorithm used for automatic citation screening. Following previous approaches [10,11], we use a Multi-Channel CNN using varying kernel sizes to classify a citation's relevance to the review.

In van Dinter, Catal and Tekinerdogan [12], we showed that the Multi-Channel CNN is applicable in a Decision Support System. As a succession, this study assesses our neural network-based document classification method's performance by investigating six architectures using a different number of channels and varying kernel sizes. For evaluation, we conducted a series of experiments to investigate our approach's performance when applied to the citation screening task of 20 publicly available systematic review benchmark datasets from the medical domain [5, 13]. Experimental results validate that our proposed method can reduce the number of items that need to be manually screened without decreasing the review's sensitivity, i.e., at least 95% of relevant studies are identified by the semi-automatic screening method. Moreover, our Multi-Channel CNN approach shows substantial performance improvements compared to two out of 20 benchmark datasets.

Our contributions to reduce human workload and increase WSS (Work Saved over Sampling) for primary study selection processes are as follows:

- We have developed a binary text classification using Multi-Channel Convolutional Neural Networks to support the citation screening process in systematic literature reviews.
- We have evaluated our architecture across 20 systematic review datasets from the medical domain to evaluate the proposed method's effectiveness.
- Our citation screening method yields significant workload savings of at least 10% in 18 out of 20 review datasets.
- Our project is publicly available and open source at https://github.com/rvdinter/multichannel-cnn-citation-screening.

The following sections are organized as follows: Section 2 presents the related work and background. Section 3 discusses the methodology. Section 4 shows the results. Section 5 presents the discussion. Section 6 explains the conclusion and future work.

2. Background and related work

2.1. Background

2.1.1. Systematic literature reviews

A systematic literature review (also known as a systematic review) is a means of identifying, evaluating, and synthesizing all available research relevant to a particular research question, or topic area, or phenomenon of interest [1]. An SLR's goal is a reliable and rigorous method to gain clear, reasonable, and unbiased information on a research topic [14].

James Lind is seen as the first doctor to conduct SLR as we know it now. In his article *Treatise of the Scurvy* (1753), he conducted systematic clinical trials of potential cures for scurvy-trials in which oranges and lemons came out as decisive winners [15]. From that point, SLR became an extensively used practice to support evidence-based medicine. The success of SLR in evidence-based medicine triggered various other research areas to adopt similar SLR approaches [14]. In 2007, Kitchenham attempted to construct guidelines for performing SLR that satisfied the needs of software engineering researchers [1]. Since this moment, software engineering researchers widely use the SLR method to conduct unbiased research—this thesis aimed to identify and evaluate the evidence regarding natural thermal process flavors. Therefore, an SLR is a suitable research method for this thesis.

Table 1 lists the steps in the systematic review process, as proposed by [1]. Synonyms that were used in the literature were noted for consistency.

Citation screening, the sixth step in the systematic literature review, is known as the most time-consuming step in the process [16–18]. During this step, reviewers aim to exclude irrelevant citations from the review while keeping relevant citations. Citation screening is often achieved by first reading all titles, reading the abstracts, and finally, the full document. To have the least bias, reviewers use selection criteria to which the documents must match. These criteria have been set up while *Developing the Review Protocol* (SLR3). As the study selection is a binary classification procedure (i.e., annotating whether an article is *included* or *excluded*), we can leverage machine learning and NLP techniques to reduce its time consumption.

2.1.2. Machine learning

Machine learning is based on the challenge of finding patterns in data through statistics. In our use case, these statistical algorithms aim to discriminate between irrelevant and relevant citations for the SLR study. Shallow machine learning algorithms, such as logistic regression, support vector machines, and decision trees for text classification are often based on the popular two-step method. As Minaee, Kalchbrenner, Cambria, Nikzad, Chenaghlu and Gao [19] describes, this two-step method relies on (1) the development of hand-crafted features from textual documents (i.e., Natural Language Processing (NLP) preprocessing and representation techniques) and (2) the classification of these features. However, the two-step method has four significant drawbacks, as shown in the left column of Table 2.

However, opposed to shallow machine learning, deep learning relies on many layers creating a complex and flexible network. Through this flexibility, deep learning algorithms can extract feature representations without needing a domain expert. In the right column of Table 2, more promises of deep learning are provided that can cope with the drawbacks of shallow learning.

We want to add that deep learning models require high amounts of data, which is a bottleneck for some current systematic literature studies. However, each year, more and more studies are being published. In this big data context, hand-crafted feature engineering lacks scalability, requiring deep learning algorithms to learn the features directly.

A recent study by Minaee, Kalchbrenner, Cambria, Nikzad, Chenaghlu and Gao [19] reviewed over 150 deep learning frameworks proposed for various text classification problems, providing a rationale for using model architecture. We have adopted their generalized description of deep learning models into categories based on their main architectural contributions to Table 3.

CNNs and RNNs are the mainstream architectures to achieve remarkable results, even though they accomplish these results through very different approaches. The following listing provides a comprehensive overview of these two key architectures:

1. Convolutional Neural Network (CNN): Four types of layers are applied in CNN models, namely convolutional layers, pooling layers, dropout layers, and fully-connected layers. Convolutional layers use filters, which can be seen as the neurons of the layer, calculating an output value (i.e., feature map) based on the weighted inputs [23]. Pooling layers are used to down-sample patches of the feature map, making the model robust against local translations [20]. In a typical CNN model, convolutional layers are followed by a pooling layer, which is repeated (e.g., up to 152 times for the ResNet architecture [24]), and finally, a fully connected layer is applied.
2. Recurrent Neural Network (RNN): RNN algorithms use sequential information in the network [25]. RNNs aim to overcome the main challenge for CNNs: knowledge about the past. In examples like predicting the upcoming word in the sentence, past words are needed, and subsequently, it is required to remember past words. The RNN illuminated this problem with the assistance of a Hidden Layer. The principle and most significant component of RNN is the Hidden state, which recollects some sequence data in internal memory. Long-Short Term Memory (LSTM) models are the most popular RNN algorithms [10,25].

The main catch here is that: (1) Texts can be seen as sequences, and RNN architectures have been widely used for sequence classification, (2) RNNs can obtain contextual information, but the order of words also results in bias, (3) CNNs can obtain essential text features from spatial input, such as keywords and sentences, as they excel at learning the spatial structure in input data, but it is challenging to grasp contextual information [26].

In the text classification field, researchers often mention the use of a Bidirectional layer for RNN architectures. Bidirectional approaches are used to process the input text, storing the previous and future tokens' semantics. Furthermore, when using a CNN, researchers frequently opt for a Multi-Channel approach. The model loads the source document using different kernel sizes, producing a Multi-Channel CNN that reads the text with various n-gram sizes [10].

For a few years, studies are combining the strengths of CNNs and RNNs, by cascading the two architectures into a hybrid model [27–29], called C-LSTM (CNN-LSTM) [29]. This hybrid architecture concept uses CNN layers to retrieve a sequence of higher-level phrase representations and is fed into an LSTM to acquire contextual information from the local phrase representation. [27] managed to produce a model with the highest reported accuracy on the IMDB review sentiment dataset with this cascaded architecture. [11] have proposed several other variations of a hybrid model architecture using CNN and LSTM. They proposed an LSTM cascaded with a CNN, which aims to extract the critical local features from the LSTM layer's output. At last, they proposed a model architecture that concatenated the outputs from an LSTM and CNN. This model architecture aims to provide both contextual and local features from the text.

2.2. Related work

Prior work on automating the primary study selection process shows that full-text document features are often avoided, as full-text articles are significantly different from titles and abstracts. Furthermore, full-text articles need much textual cleaning, and sometimes conversion from PDF to text, and access to full-text articles is a problem [30–32]. These challenges are easily avoided by using, for instance, the title and abstract, which Dieste and Padua [33] confirm. Besides, titles and abstracts are more often available for extraction through APIs, a method which Langlois,

Table 1
Steps in the systematic review process as proposed by Kitchenham and Charters [1].

ID	Category	Step	Synonyms
SLR1	Need for a review	Commissioning a review	
SLR2		Specifying the research question(s)	
SLR3		Developing a review protocol	
SLR4		Evaluating the review protocol	
SLR5	Conducting the review	Identification of research	Literature Search, Search String Development
SLR6		Selection of primary studies	Citation Screening
SLR7		Study quality assessment	Selection Review
SLR8		Data extraction and monitoring	
SLR9		Data synthesis	
SLR10	Reporting the review	Specifying dissemination mechanisms	
SLR11		Formatting the main report	
SLR12		Evaluating the report	

Table 2
Shallow learning for text classification drawbacks and its corresponding deep learning promise.

No.	Shallow learning drawbacks	Deep learning promises
1	Hand-crafted features require to be heavily fine-tuned to achieve good performance, which is a tedious and time-consuming task and must be performed by an expert [19]	Deep learning methods can learn feature representations from natural language that are required by the model [10]
2	Models are heavily fine-tuned on a few domains; it is challenging to generalize new domains [19]	Deep learning models and learned features can be reused for new domains and sometimes for whole new tasks through transfer learning [20]
3	Features are pre-defined; these models cannot take full advantage of vast training data volumes [19]	By also learning feature representation, deep learning models can discover hidden data patterns and be applied to other domains [19]
4	In a pipeline of hand-crafted models, each piece requires specialized expertise [10, 20]	An end-to-end model adds speed and simplicity of development, as it is a more general approach

Table 3
Models categorized based on architectural contributions, as adopted from Minaee, Kalchbrenner, Cambria, Nikzad, Chenaghlu and Gao [19].

Abbreviation	Model architectures	Description	Pros	Cons
RNN	Recurrent neural network	View text as a sequence of words and are intended to capture word dependencies and text structures	High overall accuracy potential [21]	Takes high training and execution time compared to CNN and Memory Augmented Neural Network (MANN) [21]
CNN	Convolutional neural network	Trained to recognize patterns in text, such as key phrases, for classification	Fast computation as it can execute in parallel	Max-pooling may result in losing important information. CNNs are also not able to find relationships between local features [21]
CapsNet	Capsule neural network	Address the information loss problem suffered by the pooling operations of CNNs, and recently have been applied to text classification	Can be trained with much less information than other neural network-based architectures [21]	Are not able to generalize well on complex datasets, such as CIFAR-10 [22]
	Attention mechanism	Useful to identify correlated words in a text and has become a useful tool in developing deep learning models	Better results than RNN, as it decides which part of the text to focus on [21]	Takes high training and execution time compared to CNN and MANN [21]
	Hybrid models	Combine attention, i.e., RNNs, CNNs to capture local and global features of sentences and documents	Combines the pros of RNNs and CNNs	Takes high training and execution time compared to CNN and MANN [21]

Nie, Thomas, Hong and Pluye [34] and Rúbio and Gulo [35] discussed the need for.

The main NLP preprocessing steps for selecting primary studies are the removal of stop words and stemming. Also, Bag of Words (BoW) and Term Frequency–Inverse Document Frequency (TFIDF) techniques are the main NLP representation. [36–39] used word embeddings, which are numerical feature representations that allow words or sentences with similar meanings to have an equal representation [10]. Similar representations resolve the need for extensive sentence cleaning, such as lemmatization and stemming. Furthermore, Brownlee [10] describes that: "The use of word embeddings over other text representations is one of the key methods that has led to a breakthrough performance with deep neural networks on problems like machine translation". [10]

Supervised machine learning is the primary technique for the automation of primary study selection. The main evaluation metrics used are precision, recall, and F-measure, but the primary metric in this field is Work Saved over Sampling (WSS). As shown in Eq. (1), WSS was founded by Cohen, Hersh, Peterson and Yen [5]. As they describe, "We define the work saved as the percentage of papers that meet the original search criteria that the reviewers do not have to read (because they have been screened out by the classifier)". [5].

$$WSS = \frac{TN + FN}{N} \tag{1}$$

TP is the number of true positives, TN is the number of true negatives, FN is the number of false negatives, N is the total number of abstracts in the set. Cohen, Hersh, Peterson and Yen [5] stated that one should interpolate the WSS metric at a 95% recall, as work saved must be greater than work saved by plain random sampling [5]. Eq. (2) shows the formula when incorporating recall R in the formula.

$$WSS@R = \frac{TN + FN}{N} - (1 - R) = \frac{TN + FN}{N} - \left(1 - \frac{TP}{TP + FN}\right) \tag{2}$$

The major challenge that studies encounter during the automation of the citation screening process is class imbalance [4,17,37–45]. This challenge exists since the citation screening process often must deal with a skewed distribution of a high number of negatives and a small number of positives. Such a skewed distribution causes classification problems, as most classifiers tend to maximize overall accuracy. [31,46] highlight the need to select the best features for their models. In addition, Cohen, Ambert and McDonagh [47] suggested using a ranking model instead of an Active Learning model as reviewers like to hold control over the results.

It has taken until early 2020 for a paper to use Deep Learning algorithms to automate the citation screening process [39]. In their paper, Kontonatsios, Spencer, Matthew and Korkontzelos [39] describe that they have used a denoising autoencoder combined with a deep neural network for document feature extraction. Consecutively, they used weights from input and output layers in the feed-forward network as input for an SVM to classify relevant primary studies. In conclusion, to this day, there is still no study that uses Deep Learning algorithms to select primary studies in the SLR process.

In terms of Deep Learning algorithms, Multi-Channel CNN architectures have become more and more used for NLP tasks. Researchers leverage the speed and low computational cost of CNN models while maintaining high scores. In 2014, Yoon [48] presented a paper that compared several CNN and shallow learning algorithms, one of which was the Multi-Channel CNN. The models were compared against seven benchmark datasets, which consisted of binary and categorical text classification. The Multi-Channel CNN was able to outperform on the *Stanford Sentiment Treebank* and *Customer Reviews of Products* datasets.

Additionally, Colón-Ruiz and Segura-Bedmar [11] performed a study comparing deep learning architectures for sentiment analysis of drug reviews. This approach is rather similar to our study, as it used benchmark datasets from the Medical domain. However, the datasets are much larger, and thus the models are much less likely to overfit. [11] compared several CNN, LSTM, and hybrid architectures. Furthermore, [11] mentioned that *"CNN networks are good at extracting local and location-independent features, but they are not able to extract information from long-range semantic dependencies"* [11]. [11] also showed that the simple CNN model could train as fast as 23 s, while the LSTM model took up to 1383 s.

Finally, in his guidelines, Brownlee [10] explains how to implement a basic Multi-Channel CNN architecture for NLP problems. As [10] explains: *"The model can be expanded by using multiple parallel convolutional neural networks that read the source document using different kernel sizes. This, in effect, creates a Multi-Channel convolutional neural network for text that reads text with different n-gram sizes (groups of words)"* [10].

3. Research methodology

This section discusses the development of the automated study selection framework. First, we discuss the Multi-Channel CNN design, such as the word embeddings we have used as a vector representation that will be fed to the embedding layer of the neural network, then we discuss the evaluation metric we have used, and last, we discuss the adopted architectures. Then, we discuss the case study design and the datasets we have used to evaluate our model.

3.1. Design of the multi-channel CNN

3.1.1. Feature length

As many citations (e.g., interviews or book chapters) that occur in the systematic review process do not contain an abstract, we concatenate the title and abstract to eliminate empty features. We have plotted the feature column's length in a probability density function and cumulative distribution function, as shown in Fig. 1. Here, we have combined all citations across the review datasets discussed in Section 3.2.1 to obtain a reliable average. In the probability density function, we see a spike at about 30 and 300. The first spike is when a citation contains only the title. The second spike indicates that most articles with a title and abstract have a total text length of approximately 300.

3.1.2. Word embeddings

Traditional natural language representation methods, such as Bag of Words, depend on one-hot encodings. In this representation, each word is represented by a one-bit position in a vector of the vocabulary length, which could be thousands long. Furthermore, this traditional method does not leverage context information but just word frequency.

Therefore, we use Stanford University's pre-trained Global Vectors for Word Representation (GloVe) word embeddings [49]. GloVe word embeddings are gained from unsupervised training on a large dataset, retrieved from websites such as Wikipedia or Twitter. The GloVe word embeddings enable the Multi-Channel CNN to look further than just word frequency; it enables the algorithm to search for similar key words or sentences.

In word embeddings, words are represented by a vector of a fixed number of dimensions (i.e., generally 50, 100, or 300), and similar words have similar representations [10]. The Euclidean distance between two individual word vectors provides a method for measuring the similarity between words. The GloVe algorithm is an extension to the Word2Vec method, as it aims to profit from both Word2Vec's local context-based learning as well as matrix factorization techniques [10]. One more drawback is the file size of Word2Vec pre-trained embeddings, which is significantly larger than GloVe embeddings [49,50].

Additionally, as the GloVe word embeddings aim to provide a similar representation for similar words, it allows for very minimal text cleaning. We use its most common representation through the Wikipedia embedding dataset, containing 6B tokens, a 400 K-sized vocabulary, and 100-dimensional vectors. We used the GloVe embeddings to create an embedding matrix that contains a word embedding for each word occurring in the dataset [10]. Eq. (3) represents the format of an embedding matrix.

$$E = \begin{bmatrix} e_{11} & \cdots & e_{1d} \\ \vdots & \ddots & \vdots \\ e_{w1} & \cdots & e_{wd} \end{bmatrix} \tag{3}$$

Here, E is the embedding matrix that contains w words. Each word is represented by a vector e, which has a length of d, which is now 300. Later, we can use this embedding matrix as an Embedding layer to learn jointly with the neural network. However, the pre-trained word embeddings are static, as the embeddings are a good fit for our problem, while it reduces computing cost [10].

3.1.3. Steps per epoch

To avoid the model from jumping around the search space, one needs to take a large batch size, preferably as large as a GPU can handle, but it does not get stuck in local minima. However, a smaller batch size could help with adding noise when handling with little data, thus generating a more robust model [51].

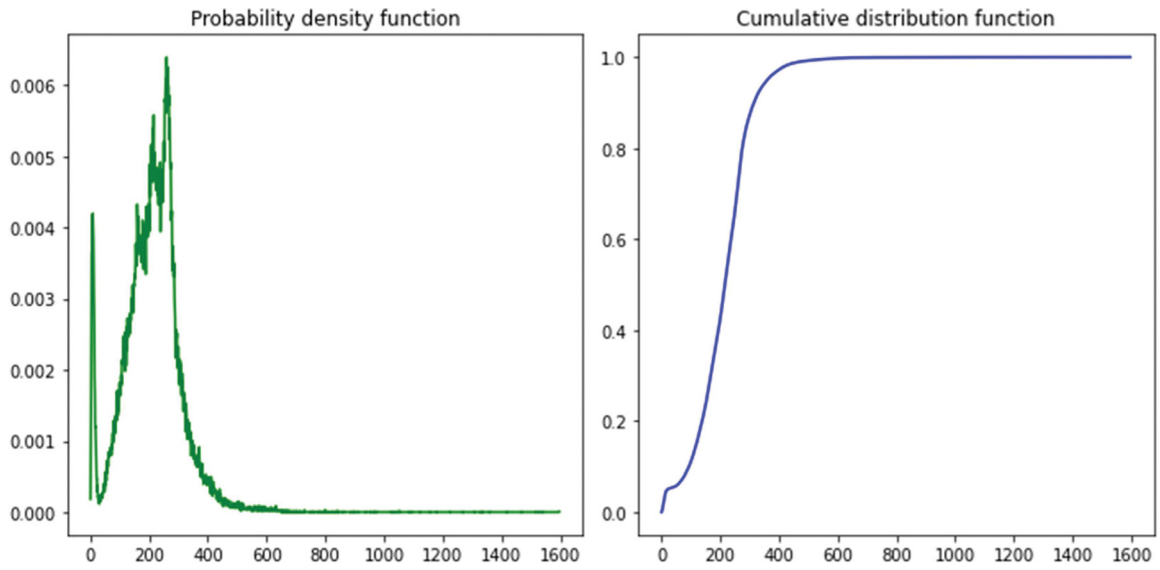

Fig. 1. Probability density function and cumulative distribution function of all 20 datasets, with title and abstract combined into one text feature column.

At last, we calculate the steps per epoch, i.e., the number of batches required to see each negative sample once, using Eq. (4), which is adapted from [52].

$$S_{epoch} = \left\lceil \frac{N}{B_{size}} \right\rceil \tag{4}$$

Here, S_{epoch} is the steps per epoch, N is the number of training samples in the set, and B_{size} is the batch size. As we are oversampling our training set to a 50/50 distribution, we want to know the steps per epoch when seeing all negative samples at least two times, shown in Eq. (5).

$$S_{epoch} = \left\lceil 2 * \frac{N_{neg}}{B_{size}} \right\rceil \tag{5}$$

Here, N_{neg} is the number of negative training samples.

3.1.4. Evaluation settings

For each of the model evaluations, we show the WSS@95% results. To be more specific, we show the mean WSS@95% after 10 × 2-fold cross-validation. 2-fold cross-validation splits the dataset into two equally sized subsets, with an even distribution of label classes. Like [5,7,13,53], we chose 2-fold cross-validation, as choosing a higher number of folds would also result in a more extensive training set in the real world. We performed this 2-fold cross-validation over ten fixed seeds to achieve a final estimated mean.

The related work section shows multiple other evaluation metrics. However, as Ng [54] described, adding more than one metric also makes it more complex to compare algorithms. As [54] describes: "Having a single-number evaluation metric such as accuracy allows you to sort all your models according to their performance on this metric, and quickly decide what is working best". [54]. Therefore, we keep WSS@95% as our single metric, as it measures the work saved while still retaining as many citations as possible.

3.1.5. Preprocessing

Once we have loaded the dataset, we concatenate the title and abstract into one feature column; we clean the text by splitting the text into tokens, removing its punctuation, converting to lower case, removing non-alphabetic and stop words, removing short tokens of just one character, and applying a minimal token occurrence of 10 in the full dataset.

After cleaning the sets, we split the dataset into a stratified train, test, and validation set for our 10 × 2 cross-validation. These sets have been split into a 45/50/5 distribution, respectively. We used the validation set to monitor the training process during cross-validation. Further, we used the Tokenizer API to create numeric word vectors from the feature column. We zip the feature and target columns into a tf.data.Dataset object. Using this object, we can oversample the training set using tf.data.experimental.sample_from_datasets(), to an equal class distribution to avoid class imbalance issues. Once the datasets are complete, we pad and truncate the datasets to a maximum length of 600, as 99.77% of all tokens remain included when truncating the text feature to 600.

3.1.6. Avoidance of overfitting

Smaller weights in the neural network may result in a more robust and less likely model to overfit on the training dataset, which would increase performance when generalizing on the test set [51]. To account for these smaller weights, we use weight constraints. Compared to weight regularization, a weight constraint is a trigger that measures the size or magnitude of the weights and adjusts them so that they are all below the predefined threshold. Weight constraints limit weights to a threshold and can be used instead of weight decay and in combination with more aggressive network configurations, such as high learning rates or when datasets are sparse [51]. We are using the unit norm to force weights to have a magnitude of 1.0.

Furthermore, it is known that deep neural networks are likely to overfit when training on sparse datasets. Therefore, we use dropout. Using dropout, a single model can simulate having many different network architectures (e.g., ensemble architectures) by randomly dropping out nodes during training. Dropout is a very computationally cheap and remarkably effective regularization method to reduce overfitting and generalization errors in deep neural networks of all kinds [51].

3.1.7. Deep learning architectures

We have considered six different Multi-Channel CNN architectures to evaluate their effectiveness of document classification in systematic literature reviews, as shown in Table 4. We use Multi-Channel CNN architectures as they are faster and computationally less expensive than LSTM architectures. To date, using large LSTM architectures is not available to all researchers without access

Table 4
Six different model configurations using varying channels and kernels.

Name	# of channels	Kernel sizes
Model_1	2	2/3
Model_2	2	2/4
Model_3	3	2/3/5
Model_4	3	2/4/6
Model_5	4	2/3/5/7
Model_6	4	2/4/6/8

to GPU hardware with high memory. However, CNN models allow for impressive results with fewer hardware requirements and lower training times. As we mentioned in the Background section, CNNs also focus on finding keywords and phrases in text classification, which researchers often do in the SLR process while skim-reading many articles.

Fig. 2 shows an abstraction of the Multi-Channel CNN architecture. All models use the text feature column as input and one binary output. An Embedding layer follows the input to create word embeddings in an end-to-end fashion. We have used the GloVe embedding matrix as input for our embedding layer. The embedding layer does not need to train its parameters, as we have inserted a pre-trained embedding matrix, which significantly reduces the training time for the model. For each of the CNN channels, we use a single CNN layer followed by global max pooling. After the global pooling layer, we concatenate the outputs and put them into a feed-forward network. The hidden dense and convolutional layers use the ReLu activation function to avoid the vanishing gradient problem. The last dense layer uses a Sigmoid activation function to account for binary classification.

We have kept a logbook of all hyperparameter settings and have put the best model parameters based on the Statins set in Table 5. Next to these model parameters, we have experimented with different channels and kernel sizes, which can be seen in Table 4. Kernel sizes and the number of channels are variations from Model_3 based on similar parameters adapted from [10,11].

From the baseline models gained from [10,11], we gathered the main hyperparameters to tune, as mentioned by [10,51]. When using fewer channels, we cut off the largest kernel sizes, thus incorporating less phrase information and emphasizing keywords. Furthermore, we also defined hyperparameters that were not actively tuned. As mentioned by [51], we implemented a unit norm bias and kernel constraint for the dense layers to reduce overfitting. For the optimizer, we used the renowned Adam algorithm together with the binary cross-entropy loss function.

The number of filters provides many features for the dense layers, while dropout prevents the model from overfitting the training data. Furthermore, reducing the batch size allows for faster training and adds noise to create a robust model. The current learning rate allows a smooth learning curve, while the number of epochs defines the number of iterations that data passes through the model.

Additional research could focus on other parameters, such as momentum, regularizers for the convolutional layers, and optimizers such as the Focal Loss Optimizer.

3.2. Case study setup

3.2.1. Datasets

We have collected 20 publicly available datasets from [13] and [5] to evaluate our model. These datasets have been regularly used to evaluate models in the medical domain. We have also collected the WSS@95 results from [5,7,13,39,53] as benchmarks for evaluating our results. Table 6 shows the metadata for each of the datasets. Each sample (i.e., citation) contains the title, abstract, and label. The 15 datasets from [5] can be categorized as drug reviews, while five datasets from [13] are categorized as SWIFT reviews. The SWIFT reviews are substantially more extensive than the drug review sets, as the researchers used broad search strategies. To tune the hyper-parameters of our method, we used one development review, namely the Statins dataset that consists of 3463 samples. From the datasets, an average of approximately 5.2% of abstracts is missing. However, this differs significantly between datasets. For instance, the Neuropathic Pain dataset has 0 abstracts missing, but the Statins dataset has 20.82% of its abstracts missing.

3.2.2. Case study design

We designed the case study to simulate best the system's performance, shown in Fig. 3. First, we needed to gather the review datasets by Howard, Phillips, Miller, Tandon, Mav, Shah, Holmgren, Pelch, Walker and Rooney [13] and Cohen, Hersh, Peterson and Yen [5]. Second, as the review datasets consisted only of PubMed IDs, we used the PubMed API to collect the citations' title and abstract iteratively. Other metadata was also included in the final dataset but not utilized. Third, we load the GloVe word embedding file. We later use the GloVe embeddings in the training phase to create an embedding matrix of the vocabulary utilized in the embedding layer. Fourth, we preprocess the dataset. We use NLP techniques mentioned before and concatenate the title and abstract into a single text feature column. Fifth, we split the preprocessed dataset into a train, test, and validation set. Even though the validation set is not of critical use in evaluating these models, we remain using the split, as true to the real-world usage. Sixth, we train the Multi-Channel CNN model. Last, we evaluate the Multi-Channel CNN models using the WSS@95% metric.

4. Results

This section discusses the model architectures' results, the number of epochs, and a further explanation of analyzing training scores for oversampled datasets.

4.1. WSS@95% for oversampled train sets

WSS@95% is a metric specially developed for systematic literature review automation systems. During the development of this metric, Cohen, Hersh, Peterson and Yen [5] focused on the imbalanced nature of SLR datasets. However, when oversampling the training set, the cross-validation results for this set are skewed. Due to the nature of WSS@95%, which formula can be found in Eq. (1), WSS@95% can only be high when the number of negatives is high as well. We will take two datasets as an example, one balanced, one imbalanced.

Example 1 (*Imbalanced Dataset*)**.**

$$N = 2000, \quad Positive = 100, \quad Negative = 1900$$

We want to interpolate when the recall is approximately 95%. Assume that in the most optimal situation, we have predicted all positives correctly.

$$R = \frac{TP}{TP + FN} = \frac{100}{100 + 5} \approx 0.95 \tag{6}$$

Then, in the most optimal situation, we would also have no False Positives. This means that we would have 1895 True Negatives. This would result in a WSS@95% of:

$$WSS@95\% = \frac{TN + FN}{N} - (1 - R) = \frac{1895 + 5}{2000} - 0.05 = 0.90 \tag{7}$$

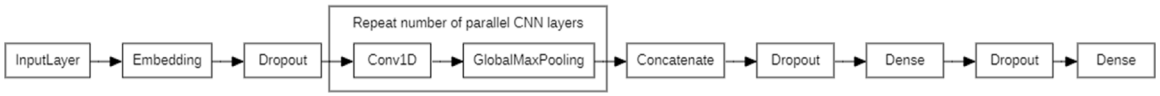

Fig. 2. Abstraction of the Multi-Channel CNN architecture. The Conv1D and GlobalMaxPooling layers will be added in parallel, indicated by the number of channels.

Table 5
Model hyperparameter settings.

Epochs	Batch size	Dropout input layer	Dropout hidden layers	Filters	Dense units	Learning rate
15	100	0.6	0.4	1024	128	1E−4

Table 6
Datasets adopted by Howard, Phillips, Miller, Tandon, Mav, Shah, Holmgren, Pelch, Walker and Rooney [13] and Cohen, Hersh, Peterson and Yen [5].

Author	Dataset	# Citations	Eligible citations (%)	Missing abstracts (%)
Howard, Phillips, Miller, Tandon, Mav, Shah, Holmgren, Pelch, Walker and Rooney [13]	Bisphenol-A (BPA) and obesity	7700	1.44	7.88
	PFOA/PFOS and immunotoxicity	6328	1.50	5.97
	Transgenerational inheritance of health effects	48 638	1.57	4.38
	Fluoride and neurotoxicity in animal models	4479	1.14	13.60
	Neuropathic pain	29 202	17.2	0.00
Cohen, Hersh, Peterson and Yen [5]	Angiotensin-converting enzyme (ACE) inhibitors	2544	1.64	12.15
	Attention deficit hyperactivity disorder (ADHD)	851	2.35	5.64
	Antihistamines	310	5.16	7.41
	Atypical Antipsychotics	1120	13.04	7.95
	Beta Blockers	2072	2.03	9.60
	Calcium Channel Blockers	1218	8.21	9.03
	Estrogens	368	21.74	5.16
	NSAIDs	393	10.43	8.91
	Opioids	1915	0.78	7.47
	Oral Hypoglycemics	503	27.04	5.57
	Proton Pump Inhibitors	1333	3.83	9.15
	Skeletal Muscle Relaxants	1643	0.55	17.71
	Statins	3463	2.45	20.82
	Triptans	671	3.58	11.48
	Urinary Incontinence	327	12.23	13.15

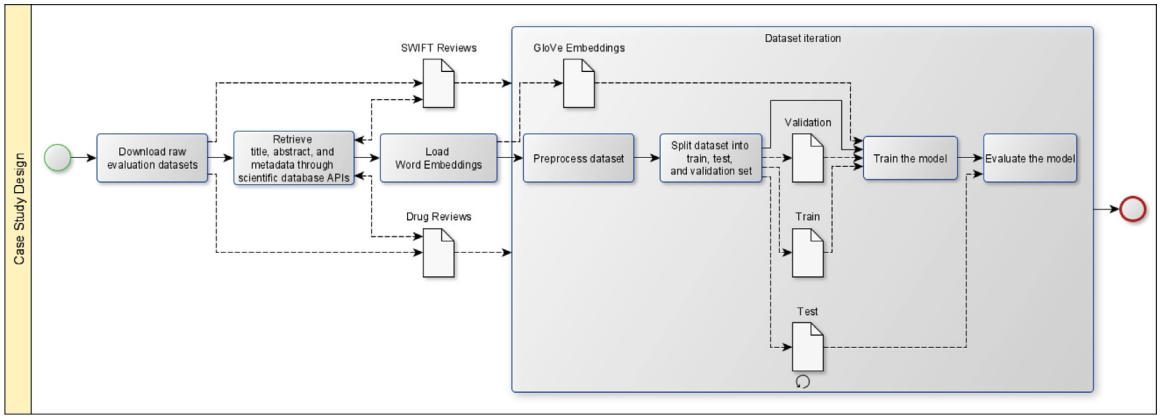

Fig. 3. The case study design.

Example 2 (*Balanced Dataset*)**.**

$$N = 2000, \quad Positive = 1000, \quad Negative = 1000$$

We want to interpolate when the recall is approximately 95%. Assume that in the most optimal situation, we have predicted all positives correctly.

$$R = \frac{TP}{TP + FN} = \frac{1000}{1000 + 50} \approx 0.95 \tag{8}$$

Then, in the most optimal situation, we would also have no False Positives. This means that we would have 950 True Negatives. This would result in a WSS@95% of:

$$WSS@95\% = \frac{TN + FN}{N} - (1 - R) = \frac{950 + 50}{2000} - 0.5 = 0.45 \tag{9}$$

We can see that only an imbalanced set can achieve a high WSS@95% score in the ideal situation. This means that this metric has a different scaling as a train set than a validation or test set.

4.2. Number of epochs

Fig. 4 shows the WSS@95% results for our development set. The uninterrupted lines represent the mean of 5-fold cross-validation, while the confidence bands show the 95% confidence interval. We can see that the training set stops to improve at 0.5, and the validation set converges at that same number. The test set's results were only analyzed after training the model, as it was our hold-out set. After 5-fold cross-validation, the green horizontal line indicates the mean of the test results. If we would

the model train for longer, chances of overfitting would increase, as the validation set's results tended to decrease after training for more than 15 epochs. Therefore, we have set the number of epochs for all datasets to 15.

4.3. Effect of model architectures

In this sub-section, we provide insights into the performance of our model architectures. Table 7 shows the WSS@95% results of our six models. We can see that the average score for Model_2 is highest on the SWIFT review datasets by [13], while Model_5 achieved the highest average score on the drug review datasets by [5]. Furthermore, Model_2 achieved statistically significant higher scores for 3 out of 5 SWIFT review datasets. We can see that increasing the number of channels or kernel sizes – while fixing the other – is not directly causing higher scores.

Fig. 5 shows the architecture of Model_2. The main difference with the other models is the number of channels (NB Model_2 contains two channels) and the output of the Conv1D layer due to the varying kernel shapes. Here, the input layer is fed a feature sample consisting of a preprocessed title and abstract. This input is followed by the Embedding layer, which also uses Stanford's Wikipedia GloVe word embeddings. The embeddings are set to be non-trainable, as they are pre-defined, and training them would consume lots of time with little improved performance. The features are fed to the dropout layer, which drops out 60% of the features. Dropout only applies to the model when training, so the full potential can be used to generalize new data. Then, the remaining data is fed to two parallel convolutional channels. As textual data is one-dimensional, we also use the one-dimensional convolutional layers. We used 1024 filters, the ReLu activation function, and varying kernel sizes. After, the filters are fed to the global max-pooling layer. We chose the global max-pooling layer over regular max-pooling, as Jacovi, Shalom and Goldberg [55] describe: "Global max-pooling induces a functionality of separating important and not important activation signals using a latent (presumably soft) threshold" [55]. Then, we concatenate the outputs from the global max-pooling layer to retrieve a single vector. We apply a 40% dropout to the concatenated vector and apply it to the dense layer. This dense layer consists of 128 units, uses the ReLu activation function, and uses bias and kernel weight constraint following the unit norm to avoid overfitting. Then, we apply the last dropout of 40%. Last, we have a dense output layer with one unit with the Sigmoid activation function to use binary classification. This last dense layer also uses bias and kernel weight constraints.

4.4. Comparison to benchmark studies

From Table 7, we can observe that Model_2 and Model_5 achieve the highest average scores for SWIFT and drug reviews, respectively. In Table 8, we take the scores from these models and compare them to 5 benchmark studies, sorted from dated to most recent. We can see that our models outperform the benchmark studies on three datasets, Bisphenol-A, Fluoride, and Angiotensin-converting enzyme (ACE) inhibitors. We achieved a 3.6%, 1.3%, and 0.4% improvement over the BPA, Fluoride, and ACE Inhibitors datasets, respectively. Furthermore, our model performed poorly on the PFOA/PFOS dataset, as it seemed to overfit. Next to two SWIFT review datasets, the models also outperformed the ACE Inhibitor review dataset. This dataset is the #2 largest drug review dataset.

Furthermore, in Fig. 6, we have also plotted Model_2 results in boxplots against the benchmark means. We can see that most often, the benchmarks' results are inside the interquartile range (IQR). We can also observe that the Opioids dataset has an extensive range of its ten cross-validation scores. Also, the five large SWIFT review datasets have a small IQR and minimum–maximum range rather. The drug review datasets are showing a more considerable variation of IQRs and the min–max range.

5. Discussion

5.1. General discussion

This study represents the first deep learning end-to-end model for citation screening to the best of our knowledge. The results that we obtained demonstrate that our Multi-Channel CNN-based citation screening model substantially reduced the screening workload of 20 systematic reviews by approximately 41%. Model_2 performs best overall on all datasets, particularly on large datasets, while Model_5 scores best on the smaller datasets. However, this is with just a small margin. Therefore, we chose Model_2 as our final model.

The workload savings varied across the 20 reviews, from a low WSS @95% score of ∼7% on the Oral Hypoglycemics review to a higher WSS @95% score of ∼88% on the Fluoride review. The models performed poorly on the PFOS-PFOA review dataset, as it seemed to overfit. We have checked the dataset input, but it has no differences from the other datasets. Additionally, even though we used the Statins dataset as our development set, we did not manage to break the WSS@95% scores. We have kept a logbook on all model parameters we have tried, but the dataset's small size and many missing abstracts (20.82%) obstruct the neural network's performance.

Moreover, we observed a weak uphill correlation ($R^2 = 0.395$) between the WSS@95% performance and the size of the corresponding review dataset, which was statistically significant ($p = 0.085$). This indicates that our model can obtain more meaningful workload savings when the dataset size increases. Therefore, Multi-Channel CNN architectures can be adopted for the citation screening process with larger datasets. We can also conclude from looking at the IQR from the boxplots; the WSS@95% scores remain remarkably consistent with large datasets.

According to Cohen, Hersh, Peterson and Yen [5], a significant workload saving should be at least 10% for the WSS@95% metric. This stems from the fact that the citation screening process of a systematic review, when conducted manually, requires on average ∼8.7 FTE to be completed, based on a 38-hour workweek. Therefore, a WSS@95% score of 10%, i.e., 10% of correctly excluded citations +5% of incorrectly excluded citations, results in a workload reduction of ∼1.3 FTE. According to expert reviewers, this is a significant reduction of their citation screening labor. The experiments that we conducted showed that our proposed method yields significant workload savings of at least 10% in 18 out of 20 review datasets. Thus, it could be potentially used in practical application scenarios for accelerating the citation screening task of systematic reviews.

Our method's workload reduction (i.e., WSS@95% score) achieved by our method is relative to the underlying review dataset's size. For example, our Multi-Channel CNN, Model_2, obtained approximately the same WSS@95% performance of 0.78 on both the NSAIDs and the Neuropathic pain dataset. However, the Neuropathic pain dataset's validation sample consists of 14601 citations, and it is substantially larger than the validation sample of the NSAIDs dataset, which consists of 196 citations. In practice, this means that a WSS@95% score of 0.78 is equivalent to a workload reduction of 12,118 citations, which are automatically excluded from the Neuropathic pain review. In comparison, a WSS@95% score of 0.78 translates to a workload reduction of only 147 automatically excluded citations for the NSAIDs dataset.

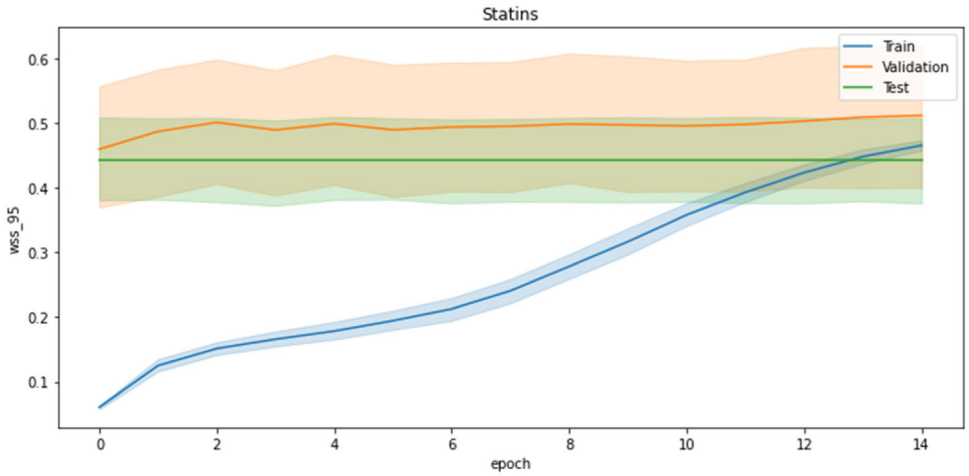

Fig. 4. WSS@95% results over 15 epochs for the Statins set.

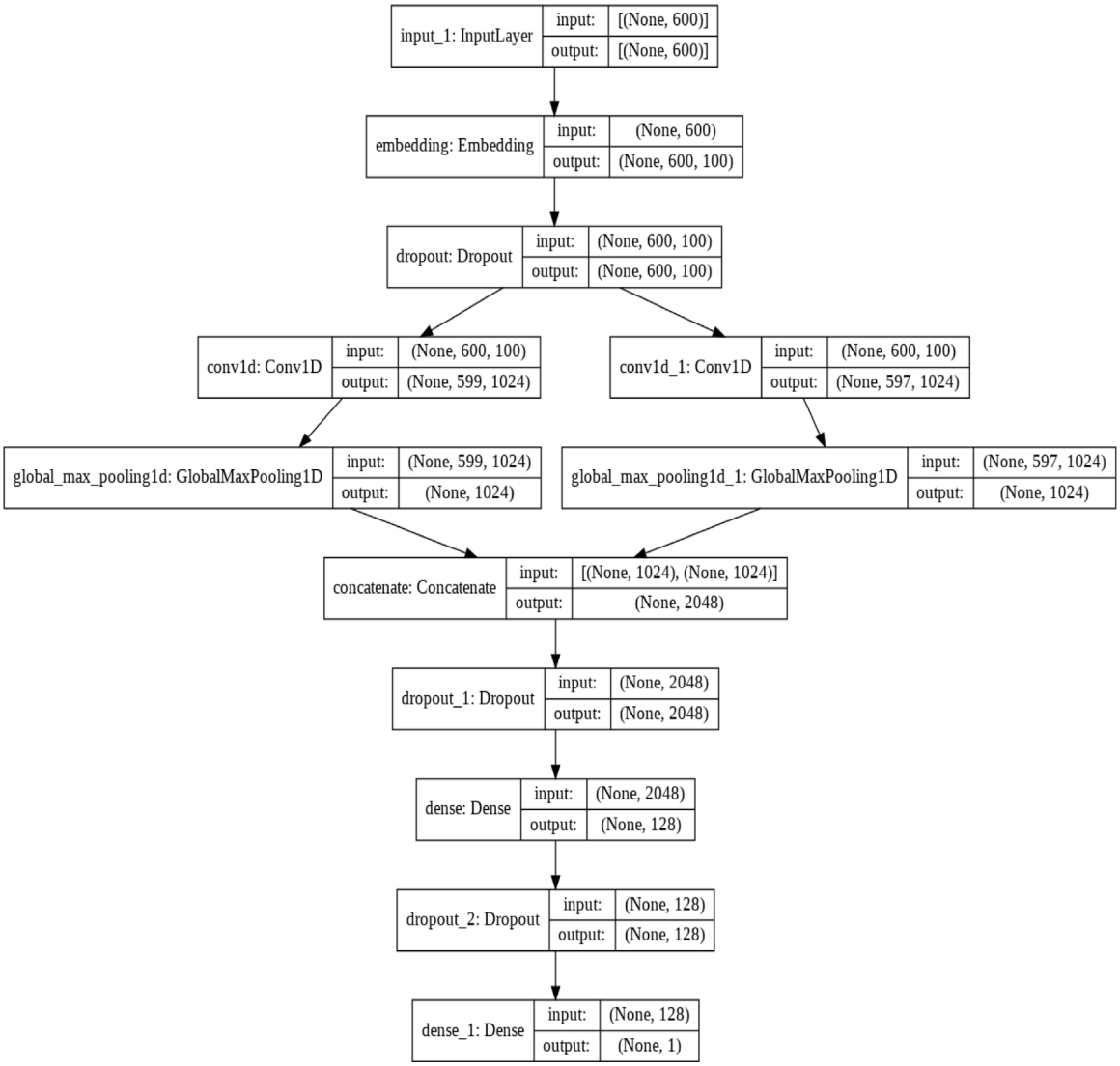

Fig. 5. Model_2 visualized. Other models are similar, except for the number of channels and Conv1D output shape due to kernel size differences.

Last, as similarly mentioned by Kontonatsios, Spencer, Matthew and Korkontzelos [39], our method's limitation is that we trained our neural network independently for each SLR dataset. This means that we have trained 20 Multi-Channel CNNs corresponding to each dataset. As [39] explains: "Different systematic reviews may share one or more eligibility criteria (e.g., if

Table 7
WSS@95% results when variating the number of channels and kernel sizes. WSS @95% scores are averages across ten validation runs for each of the 20 review datasets.

Datasets	Model_1	Model_2	Model_3	Model_4	Model_5	Model_6
BPA	0.743**	**0.792**	0.756**	0.739**	0.735**	0.719
PFOA/PFOS	0.055	**0.071**	0.054	0.057	0.059	0.048
Transgenerational	0.622**	**0.708**	0.657**	0.624**	0.592**	0.706
Fluoride	0.880	**0.883**	0.890	0.871	0.870	0.847
Neuropain	0.607**	**0.620**	0.608**	0.608**	0.610**	0.608**
SWIFT review benchmark average	0.581	**0.615**	0.593	0.580	0.573	0.586
ACE Inhibitors	0.790	**0.783**	0.797	0.793	0.805	0.811
ADHD	0.687	**0.698**	0.677	0.666**	0.683	0.669*
Antihistamines	0.150	**0.168**	0.180	0.130	0.162	0.156
Atypical Antipsychotics	0.212	**0.212**	0.220	0.226	0.234	0.232
Beta Blockers	0.506	**0.504**	0.515	0.497	0.514	0.502
Calcium Channel Blockers	0.147	**0.159**	0.155	0.158	0.177	0.181
Estrogens	0.147	**0.119**	0.116	0.173	0.171	0.149
Non-Steroidal Anti-Inflammatory Drugs (NSAIDs)	0.594	**0.571**	0.584	0.602	0.611	0.628
Opioids	0.297	**0.295**	0.306	0.297	0.290	0.286
Oral Hypoglycemics	0.052	**0.065**	0.068	0.072	0.085	0.068
Proton Pump Inhibitors	0.244	**0.243**	0.260	0.246	0.254	0.048
Skeletal Muscle Relaxants	0.222	**0.229**	0.187**	0.190	0.162**	0.156**
Statins	0.445	**0.443**	0.460	0.461	0.446	0.438
Triptans	0.257	**0.266**	0.266	0.264	0.259	0.268
Urinary Incontinence	0.257	**0.272**	0.271	0.295	0.306	0.325
Drug review benchmark average	0.334	**0.335**	0.337	0.338	0.344	0.328
Grand total	0.396	**0.405**	0.401	0.398	0.401	0.392

*Denotes statistically significant improvements over the other models at the $p < 0.05$ level.
**Shows Model_2 achieved a statistically significant better performance according to a two-tailed paired t-test over all other models at $p < 0.01$ level.

Table 8
Results of 5 benchmark studies versus the two best-performing Multi-Channel CNN models.

Datasets	Cohen et al. [5]	Matwin et al. [7]	Cohen et al. [53]	Howard et al. [13]	Kontonatsios et al. [39]	Model_2	Model_5
BPA	N/A	N/A	N/A	0.752	0.758	**0.792**	0.735
PFOA/PFOS	N/A	N/A	N/A	0.805	**0.848**	0.071	0.059
Transgenerational	N/A	N/A	N/A	**0.714**	0.707	0.708	0.592
Fluoride	N/A	N/A	N/A	0.870	0.799	**0.883**	0.870
Neuropain	N/A	N/A	N/A	**0.691**	0.608	0.620	0.610
SWIFT review benchmark average				0.766	0.744	0.615	0.573
ACE Inhibitors	0.566	0.523	0.733	0.801	0.787	0.783	**0.805**
ADHD	0.680	0.622	0.526	**0.793**	0.665	0.698	0.683
Antihistamines	0.000	0.149	0.236	0.137	**0.310**	0.168	0.162
Atypical Antipsychotics	0.141	0.206	0.170	0.251	**0.329**	0.212	0.234
Beta Blockers	0.284	0.367	0.465	0.428	**0.587**	0.504	0.514
Calcium Channel Blockers	0.122	0.234	0.430	**0.448**	0.424	0.159	0.177
Estrogens	0.183	0.375	0.414	**0.471**	0.397	0.119	0.171
NSAIDs	0.497	0.528	0.672	**0.730**	0.723	0.571	0.611
Opioids	0.133	0.554	0.364	**0.826**	0.533	0.295	0.290
Oral Hypoglycemics	0.090	0.085	**0.136**	0.117	0.095	0.065	0.085
Proton Pump Inhibitors	0.277	0.229	0.328	0.378	**0.400**	0.243	0.254
Skeletal Muscle Relaxants	0.000	0.265	0.374	**0.556**	0.286	0.229	0.162
Statins	0.247	0.315	0.491	0.436	**0.566**	0.443	0.446
Triptans	0.034	0.274	0.346	0.412	**0.434**	0.266	0.259
Urinary Incontinence	0.261	0.296	0.432	0.530	**0.531**	0.272	0.306
Drug review benchmark average	0.234	0.335	0.408	0.488	0.471	0.335	0.344

included studies are randomized control trials) and thus learned document features could be applied to different reviews".

Our study's main difference with the related work is that we have explicitly adopted a deep neural network for the citation screening process, while the only other paper used neural networks for feature extraction [39]. We have seen that the shallow machine learning architectures used domain-dependent fine-tuning of hand-crafted features in the related work. [31,46] highlight the need to select the best features for their models. The need for fine-tuning is overcome by using a practical and interchangeable NLP preprocessing pipeline combined with word embeddings. We found the key papers on the automation of the citation screening process by [5,7,13,39,53], and evaluated our results on their benchmark scores. Therefore, we have developed a deep neural network that is significantly different from shallow machine learning applications, with new and relevant insights.

5.2. Threats to validity

Construct Validity: Construct validity assesses whether the SLR represents the degree to which it measures what it asserts. First, we aimed to replicate the model by Kontonatsios, Spencer, Matthew and Korkontzelos [39], as recently published in a paper with open-source code via GitHub. However, we could not achieve the same scores using our dataset. After emailing the primary author, we were informed that he does not have access to his datasets anymore, which means their study cannot be fully replicated.

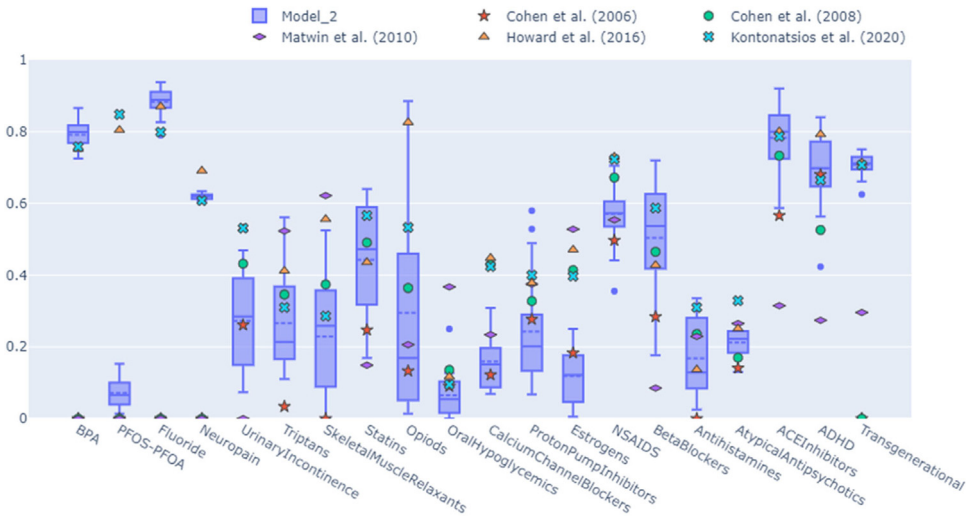

Fig. 6. WSS@95% values of Model_2 and benchmark papers. Benchmark papers WSS@95% values are shown as markers, as they are means. Multi-Channel CNN WSS@95% values are shown as boxplots. The dotted line in the boxes stands for the mean; the uninterrupted line represents the median.

Criterion Validity: To assess model WSS@95% results also during cross-validation, we have developed a TensorFlow custom metric class. As it was needed to measure WSS at a specific recall rate, we used the SensitivityAtSpecificity base class. This base class allows us to calculate a metric at another metric. We have validated this metric using our hand-written calculations. Furthermore, when validating our scores against the benchmark papers, our models seem to score in line with the other papers. However, we must note that only the work saved over citation screening is measured here, while the time of dataset construction and training is not measured through the WSS@95% metric. Nevertheless, as our Decision Support Paper describes, this model takes 7 min to train on average, and dataset construction is a matter of exporting citations to Excel and modifying the column names [12].

Internal Validity: Internal validity shows the incomplete relationship between results, which may lead to structural errors. We used cross-validation, set 10 seeds to consistently have the same dataset splits, and used fixed model hyperparameters. As these techniques were well-defined in other papers and their open-source code, the model evaluation against benchmark papers was described adequately.

External Validity: This primary study only used published studies as benchmarks that applied machine learning techniques to automate the citation screening process. The scores were required to be mentioned using the WSS@95% score, which is retrieved by *N*x2-fold cross-validation. Here *N* must be between 5 and 10 rounds. Furthermore, it is likely that a new machine/deep learning or natural language processing algorithm has not been applied yet in the automation of systematic literature reviews, like novel transformer algorithms, such as BERT and GPT-3. As these studies have not been published, they have not been discussed regardless of their potential.

Conclusion Validity: The conclusion validity measures the reproducibility of this study. Our study used datasets provided by [5,13]. Furthermore, we made our code open-source, available on this GitHub page. Our automation process was also discussed among the authors to minimize individual errors. We derived all conclusions based on the tables and figures to avoid subjective interpretation of the results among researchers.

6. Conclusion

This paper has presented a Multi-Channel CNN classification approach to support systematic literature reviews' automated citation screening process. Reviewers manually label only a subset of the citations, while our Multi-Channel CNN architecture automatically classifies the remaining unlabeled citations. This study has shown that deep learning can overcome challenges in shallow machine learning to automate the citation screening process.

We have performed six experiments to assess the performance of Multi-Channel CNNs across 20 publicly available systematic literature review datasets. It was shown that for 18 out of 20 review datasets, the proposed method achieved significant workload savings of at least 10%, while in several cases, our model yielded a statistically significantly better performance over two benchmark review datasets. We can conclude that Multi-Channel CNNs perform best on large datasets of over 2500 samples with few abstracts missing.

7. Future work

Future work could focus on the application of transformers such as BERT to automate the citation screening process. These transformers can provide contextualized word embeddings, which improve results in NLP tasks over traditional word embeddings such as GloVe [11]. Furthermore, researchers could improve our model by generating a robust model on another application domain with lots of data (e.g., IMDb movie review sentiment analysis) and use transfer learning for citation screening. Using transfer learning techniques enables researchers to leverage models trained on huge datasets and fine-tune the weights on a new dataset. As SLR datasets are often small, this approach could improve performance considerably while also reducing training times [56]. Future challenges remain the lack of data and class imbalance. The lack of data is a challenge, as large neural networks trained on small datasets can overfit the training data [51]. This challenge can be solved using transfer learning techniques or data augmentation, such as Facebook's recently published AugLy library [57]. Furthermore, the class imbalance is a challenge as most models will be biased towards the majority class [58]. Techniques such as oversampling, where samples from the minority class are shown more often to balance the majority class, or introducing loss algorithms focused on imbalanced class distributions, such as Focal Loss [59], could resolve the class imbalance challenge.

CRediT authorship contribution statement

Raymon van Dinter: Conceptualization, Data curation, Software, Writing – review & editing. **Cagatay Catal:** Methodology, Validation, Writing – review & editing. **Bedir Tekinerdogan:** Methodology, Validation, Writing – review & editing.

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.

Acknowledgment

Open Access funding provided by the Qatar National Library.

References

[1] B. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering, in, Keele University, 2007.
[2] B.C. Wallace, T.A. Trikalinos, J. Lau, C. Brodley, C.H. Schmid, Semi-automated screening of biomedical citations for systematic reviews, BMC Bioinformatics 11 (2010) 1–11.
[3] E.M. Beller, J.K.-H. Chen, U.L.-H. Wang, P.P. Glasziou, Are systematic reviews up-to-date at the time of publication? Syst. Rev. 2 (2013) 36.
[4] T. Bekhuis, D. Demner-Fushman, Screening nonrandomized studies for medical systematic reviews: A comparative study of classifiers, Artif. Intell. Med. 55 (2012) 197–207.
[5] A.M. Cohen, W.R. Hersh, K. Peterson, P.-Y. Yen, Reducing workload in systematic review preparation using automated citation classification, J. Am. Med. Inform. Assoc. 13 (2006) 206–219.
[6] J.J. García Adeva, J.M. Pikatza Atxa, M. Ubeda Carrillo, E. Ansuategi Zengotitabengoa, Automatic text classification to support systematic reviews in medicine, Expert Syst. Appl. 41 (2014) 1498–1508.
[7] S. Matwin, A. Kouznetsov, D. Inkpen, O. Frunza, P. O'Blenis, A new algorithm for reducing the workload of experts in performing systematic reviews, J. Am. Med. Inform. Assoc. 17 (2010) 446–453.
[8] O. Frunza, D. Inkpen, S. Matwin, Building systematic reviews using automatic text classification techniques, in: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, Association for Computational Linguistics, Beijing, China, 2010, pp. 303–311.
[9] R. van Dinter, B. Tekinerdogan, C. Catal, Automation of systematic literature reviews: A systematic literature review, Inf. Softw. Technol. (2021) 106589.
[10] J. Brownlee, Deep Learning for Natural Language Processing: Develop Deep Learning Models for your Natural Language Problems, Machine Learning Mastery, 2017.
[11] C. Colón-Ruiz, I. Segura-Bedmar, Comparing deep learning architectures for sentiment analysis on drug reviews, J. Biomed. Inform. 110 (2020) 103539.
[12] R. van Dinter, C. Catal, B. Tekinerdogan, A decision support system for automating document retrieval and citation screening, Expert Syst. Appl. (2021) 115261.
[13] B.E. Howard, J. Phillips, K. Miller, A. Tandon, D. Mav, M.R. Shah, S. Holmgren, K.E. Pelch, V. Walker, A.A. Rooney, SWIFT-Review: a text-mining workbench for systematic review, Syst. Rev. 5 (2016) 87.
[14] H.G. Gurbuz, B. Tekinerdogan, Model-based testing for software safety: a systematic mapping study, Softw. Qual. J. 26 (2018) 1327–1372.
[15] M. Bartholomew, James Lind's treatise of the scurvy (1753), Postgrad. Med. J. 78 (2002) 695–696.
[16] G. Tsafnat, P. Glasziou, G. Karystianis, E. Coiera, Automated screening of research studies for systematic reviews using study characteristics, Syst. Rev. 7 (2018) 64.
[17] A. Bannach-Brown, P. Przybyła, J. Thomas, A.S. Rice, S. Ananiadou, J. Liao, M.R. Macleod, Machine learning algorithms for systematic review: reducing workload in a preclinical review of animal studies and reducing human screening error, Syst. Rev. 8 (2019) 1–12.
[18] H. Sellak, B. Ouhbi, B. Frikh, Using rule-based classifiers in systematic reviews: a semantic class association rules approach, in: Proceedings of the 17th International Conference on Information Integration and Web-Based Applications & Services, Association for Computing Machinery, Brussels, Belgium, 2015, p. 43.
[19] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, J. Gao, Deep learning based text classification: A comprehensive review, 2020, arXiv preprint arXiv:2004.03705.
[20] J. Brownlee, Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python, Machine Learning Mastery, 2019.
[21] A. Yadav, D.K. Vishwakarma, Sentiment analysis using deep learning architectures: a review, Artif. Intell. Rev. 53 (2020) 4335–4385.
[22] S. Jha, Capsule Networks: A Critique, in, 2018.
[23] J. Brownlee, Deep Learning with Python: Develop Deep Learning Models on Theano and TensorFlow using Keras, Machine Learning Mastery, 2016.
[24] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[25] J. Brownlee, Long Short-Term Memory Networks with Python: Develop Sequence Prediction Models with Deep Learning, Machine Learning Mastery, 2017.
[26] M. Sawant, Text Sentiments Classification with CNN and LSTM, in, 2019.
[27] A. Yenter, A. Verma, Deep CNN-LSTM with combined kernels from multiple branches for IMDb review sentiment analysis, in: 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), IEEE, 2017, pp. 540–546.
[28] Y. Zhang, J. Zheng, Y. Jiang, G. Huang, R. Chen, A text sentiment classification modeling method based on coordinated CNN-LSTM-attention model, Chin. J. Electron. 28 (2019) 120–126.
[29] C. Zhou, C. Sun, Z. Liu, F. Lau, A C-LSTM neural network for text classification, 2015, arXiv preprint arXiv:1511.08630.
[30] K.B. Cohen, H.L. Johnson, K. Verspoor, C. Roeder, L.E. Hunter, The structural and content aspects of abstracts versus bodies of full text journal articles are different, BMC Bioinformatics 11 (2010) 492.
[31] K.R. Felizardo, G.F. Andery, F.V. Paulovich, R. Minghim, J.C. Maldonado, A visual analysis approach to validate the selection review of primary studies in systematic reviews, Inf. Softw. Technol. 54 (2012) 1079–1091.
[32] K.R. Felizardo, E.Y. Nakagawa, S.G. MacDonell, J.C. Maldonado, A visual analysis approach to update systematic reviews, in: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, Association for Computing Machinery, London, England, United Kingdom, 2014, p. 4.
[33] O. Dieste, A.G. Padua, Developing search strategies for detecting relevant experiments for systematic reviews, in: First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), IEEE, 2007, pp. 215–224.
[34] A. Langlois, J.-Y. Nie, J. Thomas, Q.N. Hong, P. Pluye, Discriminating between empirical studies and nonempirical works using automated text classification, Res. Methods 9 (2018) 587–601.
[35] T.R.P.M. Rúbio, C.A.S.J. Gulo, Enhancing academic literature review through relevance recommendation: Using bibliometric and text-based features for classification, in: 2016 11th Iberian Conference on Information Systems and Technologies (CISTI), 2016, pp. 1–6.
[36] S. González-Toral, R. Freire, R. Gualán, V. Saquicela, A ranking-based approach for supporting the initial selection of primary studies in a systematic literature review, in: 2019 XLV Latin American Computing Conference (CLEI), 2019, pp. 1–10.
[37] K. Hashimoto, G. Kontonatsios, M. Miwa, S. Ananiadou, Topic detection using paragraph vectors to support active learning in systematic reviews, J. Biomed. Inform. 62 (2016) 59–65.
[38] B.K. Olorisade, P. Brereton, P. Andras, The use of bibliography enriched features for automatic citation screening, J. Biomed. Inform. 94 (2019) 103202.
[39] G. Kontonatsios, S. Spencer, P. Matthew, I. Korkontzelos, Using a neural network-based feature extraction method to facilitate citation screening for systematic reviews, Expert Syst. Appl. X (2020) 100030.
[40] B.C. Wallace, K. Small, C.E. Brodley, T.A. Trikalinos, Active learning for biomedical citation screening, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, Washington, DC, USA, 2010, pp. 173–182.
[41] P. Timsina, J. Liu, O. El-Gayar, Advanced analytics for the automation of medical systematic reviews, Inf. Syst. Front. 18 (2016) 237–252.
[42] H. Almeida, M. Meurs, L. Kosseim, A. Tsang, Data sampling and supervised learning for HIV literature screening, IEEE Trans. NanoBiosci. 15 (2016) 354–361.
[43] O. Frunza, D. Inkpen, S. Matwin, W. Klement, P. O'Blenis, Exploiting the systematic review protocol for classification of medical abstracts, Artif. Intell. Med. 51 (2011) 17–25.
[44] G. Kontonatsios, A.J. Brockmeier, P. Przybyła, J. McNaught, T. Mu, J.Y. Goulermas, S. Ananiadou, A semi-supervised approach using label propagation to support citation screening, J. Biomed. Inform. 72 (2017) 67–76.
[45] S. Kim, J. Choi, An SVM-based high-quality article classifier for systematic reviews, J. Biomed. Inform. 47 (2014) 153–159.
[46] D.D.A. Bui, G. Del Fiol, J.F. Hurdle, S. Jonnalagadda, Extractive text summarization system to aid data extraction from full text in systematic review development, J. Biomed. Inform. 64 (2016) 265–272.
[47] A.M. Cohen, K. Ambert, M. McDonagh, Cross-topic learning for work prioritization in systematic review creation and update, J. Am. Med. Inform. Assoc. 16 (2009) 690–704.

[48] K. Yoon, Convolutional neural networks for sentence classification, 2014, arXiv.
[49] J. Pennington, R. Socher, C.D. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.
[50] Google Code Archive, word2vec, in, 2013.
[51] J. Brownlee, Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions, Machine Learning Mastery, 2018.
[52] TensorFlow, Classification on imbalanced data, in.
[53] A.M. Cohen, Performance of support-vector-machine-based classification on 15 systematic review topics evaluated with the WSS@ 95 measure, J. Am. Med. Inform. Assoc.: JAMIA 18 (2011) 104.
[54] A. Ng, Machine Learning Yearning, Deeplearning.ai (Ed.), 2017.
[55] A. Jacovi, O.S. Shalom, Y. Goldberg, Understanding convolutional neural networks for text classification, 2018, arXiv preprint arXiv:1809.08037.
[56] J. Brownlee, A Gentle Introduction to Transfer Learning for Deep Learning, in, 2019.
[57] Joanna Bitton, Z. Papakipos, AugLy: A data augmentations library for audio, image, text, and video, in, 2021.
[58] J.M. Johnson, T.M. Khoshgoftaar, Survey on deep learning with class imbalance, J. Big Data 6 (2019) 1–54.
[59] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.