Knowledge-Based Systems 284 (2024) 111286

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Optimal performance of Binary Relevance CNN in targeted multi-label text classification

Zhen Yang, Frank Emmert-Streib [*]

Predictive Society and Data Analytics Lab, Faculty of Information Technology and Communication Sciences, Tampere University, Korkeakoulunkatu 10, Tampere, 33720, Finland

ARTICLE INFO

Keywords:
Multi-label classification
Deep learning
Binary Relevance
Natural language processing
Artificial intelligence

ABSTRACT

In the context of multi-label text classification (MLTC), Binary Relevance (BR) stands out as one of the most intuitive and frequently employed methodologies. It tackles the MLTC task by breaking it down into multiple binary classification problems. However, BR has faced conceptual criticism due to its omission of label dependency information. To address this limitation, numerous studies have concentrated their efforts on enhancing the incorporation of label dependencies and document features. This resulted in substantial improvements in the performance of MLTC models. While the question of whether models incorporating label dependency information consistently outperform BR models remains unanswered, the prevailing opinion suggests their superiority. In this paper, we present evidence that challenges this widely held belief. Our numerical results across various text datasets demonstrate that an optimized binary relevance convolutional neural network (BR-CNN) can outperform advanced multi-label learning models explicitly designed to leverage label dependency information as well as advanced Binary Relevance (BR) models. Our result underscores the competitiveness of a BR-CNN approach for MLTC and emphasizes the versatility of the BR model family as a customizable option. More fundamentally, our findings contribute to the ongoing discourse surrounding label dependency and provide valuable insights into the efficacy of the binary relevance approach.

1. Introduction

Multi-label classification (MLC) has gained significant popularity, as it allows for assigning multiple labels to an input instance [1–4]. This type of classification is not only of theoretical interest but finds also practical applications in various real-world problems, including image classification, music classification, gene function classification, disease classification and document classification [5–10]. Unfortunately, MLC poses significant challenges due to several factors. First, dealing with high-dimensional data and complex correlations among labels presents difficulties for efficiently handling such information. Second, the complexity of the problem increases significantly with the number of classes. Third, the nature of the task respectively the type of data may have a significant impact on the learning model. All these challenges together make MLC a very demanding learning task, which explains the sustained interest in addressing and solving this type of problem. To make this problem more manageable, in this paper, we focus exclusively on the multi-label text classification (MLTC) task [11,12] because a method may not be universally applicable to different data types, e.g., text data, image data, audio data or sensory data.

The most intuitive approach to multi-label classification is to transform the problem into multiple binary classification problems where each binary problem is considered as an one-vs-rest classifier [13]. This reduces the difficulty of the learning task considerably and is commonly referred to as *binary relevance* (BR) classification. Unfortunately, BR approaches discard label dependency information [14,15]. In order to also utilize the label dependency information, extensions of BR methods have been proposed. One of the first methods utilizing label dependency information are classifier chains [14]. This approach uses an ordering of labels in a chain-like structure where the prediction of each label takes the predictions of all preceding labels of the chain into account. This sequential approach allows classifier chains to capture dependencies between labels and to potentially improve their performance compared to binary relevance classifiers. However, a drawback of classifier chains is that the order of the chain influences the learning behavior and the results. Furthermore, classifier chains are computationally very demanding.

In recent years, the field of multi-label classification has seen significant advancements, particularly in utilizing explicit label dependency information to improve classification outcomes. With the advent of deep learning extensive studies have been conducted, focusing on popular architectures such as Convolutional Neural Networks (CNN) [16,

[*] Corresponding author.
E-mail address: v@bio-complexity.com (F. Emmert-Streib).

https://doi.org/10.1016/j.knosys.2023.111286
Received 9 August 2023; Received in revised form 16 November 2023; Accepted 6 December 2023
Available online 15 December 2023

17], Recurrent Neural Networks (RNN) [18,19], and transformer models like BERT [20,21]. Each method tackles the challenge of capturing label dependencies using deep learning techniques from a distinct perspective. As a consequence, binary relevance (BR) methods appear to be falling out of favor, particularly in the context of deep learning-based BR methods. While some work has been done on proposing multi-label learning methods with built-in binary relevance modules, either as individual prediction layers or auxiliary predictors providing feedback to the main predictor [19], recent studies have also explored the usage of traditional classifiers. For example, Rastin et al. [22] proposed a novel training schedule for binary relevance k-Nearest Neighbor, and Kumar et al. [23] compared several BR methods based on traditional machine learning methods for movie genre classification.

These observations suggest that the utilization of deep learning-based BR methods has received limited research attention. One possible explanation for this is the significant time investment required to train multiple binary classifiers, particularly when dealing with a large number of classes. Additionally, optimizing deep learning networks can be a very challenging task. It is worth noting that deep learning architectures intrinsically possess the ability to handle multi-label learning by adjusting the final prediction layer of the network. However, at this point it is unclear whether this transition from binary to multi-label learning within deep learning architectures has a detrimental or beneficial impact on the performance.

In this paper, we aim to close this gap by investigating deep learning-based BR methods for MLTC. Specifically, we will train and optimize several commonly used deep learning architectures based on a binary relevance transformation. We compare their performance with alternative BR methods and the best performing methods utilizing label dependency information from the literature. As we will see, the optimization of the deep learning BR methods is non-trivial but necessary to achieve a competitive performance. Furthermore, in order to obtain comprehensive insights into the working mechanisms of the best performing BR architecture, we study learning curves and the influence of the number of classes. For obtaining robust results, we study four widely used benchmark datasets: Arxiv Academic Paper Dataset (AAPD) [24], Reuters-21578 [25], MIMIC-III [26] and Reuters Corpus Volume I (RCV1-v2) [27].

Overall, our study will allow to address the following five research questions:

1. What deep learning architecture gives the best BR model?
2. Can a BR method without label dependency information outperform the best performing multi-label methods utilizing label dependency information?
3. How does the performance of a deep learning-based BR method change when adjusting the prediction layer to facilitate multi-label classification?
4. Does the number of classes have a significant impact on the selection of the best performing method?
5. Do the results depend on the characteristics of the text data?

In summary, we will show that our contribution not only introduces a best performing deep learning binary relevance method called BR-CNN for MLTC by exploring multiple deep learning binary relevance architectures, but also adds valuable insights to the ongoing fundamental debate regarding the consistent superiority of methods that incorporate label dependency information over traditional BR methods. We will show that a well optimized BR-CNN can surpass the performance of the most effective methods for MLTC documented in the literature. This suggests the superiority of BR-CNN on several benchmark datasets. Our work also enhances our understanding of how to effectively leverage deep learning within the context of BR methodologies.

This paper is organized as follows. In the next section, we discuss related work about BR methods and multi-label learning methods that are the best performing methods for various datasets. In Sections 3 and 4, we discuss all methods and data we use for our analysis and in Section 5 we present our numerical results. This paper finishes with a discussion and a conclusion.

2. Related work

From the literature, one finds that binary relevance (BR) [1] is a widely used approach for multi-label text classification and one reason therefor is its simplicity. A drawback of BR is that it ignores information from label correlations which could be utilized to further improve the outcome of multi-label classification [28]. In fact, many suggest that label correlations are essential for obtaining optimal results for multi-label classification [16,24]. Consequently, substantial efforts have been devoted for exploring possibilities of leveraging label dependencies to improve the learning task [18,19,21,24]. However, despite these endeavors, a definitive guideline concerning the superiority of the usage of label dependency information for MLTC remains elusive, and it is unclear whether such methods are in general superior compared to BR methods.

Another interesting point to note is that despite the widespread adoption of deep learning [29], the utilization of deep learning-based Binary Relevance (BR) methods remains surprisingly limited. Consequently, many BR methods employed in practice rely on traditional classifiers such as Support Vector Machines (SVM) [30] or simplistic neural network architectures, which are lacking the complexity and expressiveness of more advanced deep learning models [31,32]. Unfortunately, this reliance on weaker baselines may lead to misleading conclusions regarding the effectiveness of novel methods that leverage label dependency information or deep learning architectures, as they may appear as significant advancements compared to less sophisticated BR approaches.

In the following, we will briefly review some BR methods and the best performing MLTC methods utilizing label dependency information from the literature.

2.1. Binary relevance approaches

As results from the literature show, see [33–38], deep learning methods such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) are generally good at processing textual information and have been widely applied in text classification tasks. Additionally it has been shown that CNN are good at extracting position-invariant features from the input for text classification [39]. However, there is a lack of detailed studies in the literature on optimizing the BR transformation of CNN or RNN variants. For this reason, we select CNN, LSTM and GRU as our base classifier for the deep learning BR which we optimize. We will evaluate their effectiveness on several benchmark datasets to answer our research questions. In addition, we use BR-SVM [1], BR-support [40], and BR-decoder [19] as BR baseline methods from the literature. In the following, we will elaborate on all the Binary Relevance (BR) methods employed in our analysis.

- BR-SVM: Boutell, Matthew R., et al. [1] proposed a simple way to deal with a multi-label classification tasks by decomposing the problem into multiple binary classifications using for each a linear Support Vector Machine (BR-SVM). Each class is trained for an individual BR-SVM and there are no interactions between the classes considered. Hence, if present, label dependency information among classes is lost in this way of learning.
- BR-support: Wang, Bingyu, et al. [40] proposed a training and prediction pipeline to enhance the learning of multi-label outcomes of many classifiers with respect to the F1-score. This procedure includes a support inference, a tuned and combined L1 and L2 regularization, and a so called General F-score Maximizer (GFM) for making the prediction. Their findings demonstrated that their proposed pipeline could significantly enhance the performance of multiple methods.

- BR-decoder: Tsai, Che-Ping, and Hung-Yi Lee [19] introduced a new multi-label learning architecture based on an encoder–decoder. The idea of their method is to take advantage of a RNN-based label dependency learning framework to alleviate exposure bias from a predefined order of labels by using a recursive label prediction structure. Their method consists of 3 basic components including a bidirectional LSTM-based encoder, a RNN decoder with sequential prediction of labels using an attention LSTM, and a binary relevance decoder module using a MLP. Both decoders operate individually on the encoded features from the encoder.
- CNN: Yoon Kim [41] modified the structure of a traditional convolutional neural network, making it operational for textual inputs by representing it as a two-dimensional array. Such a CNN offers an efficient way to consider features that correspond to N-grams of neighbor words. CNN can be utilized in either BR or a multi-label manner. The multi-label version is referred as ML-CNN where the output layer operates on a multi-label input by utilizing a sigmoid function and a threshold function. The BR version is referred to as BR-CNN which will be introduced in next section.
- LSTM (Long Short-Term Memory): LSTM [42] was propose by Sepp Hochreiter and Jürgen Schmidhuber and is a variant of recurrent neural networks (RNN). The term "Long Short-Term Memory" implies the capability to store information on a short term and long term basis, which is ideal for processing textual information. The four main blocks of a LSTM are cell state, input gate, forget gate, and the output gate. These elements collectively facilitate both local and long-term memory for processing inputs. Similar to CNN, LSTM can be transformed to perform multi-label learning by modifying the last prediction layer. We refer to the multi-label version of LSTM as ML-LSTM and the BR version as BR-LSTM.
- GRU (Gated Neural Networks): GRU [43] is another variant of RNN which has similar components as LSTM but with lesser parameters and gates. Both GRU and LSTM are RNN variants and are equally powerful in text classification. However, it has been shown that LSTM performs slightly better than GRU, while GRU boasts better time complexity [44]. We refer to the multi-label version of GRU as ML-GRU and the BR version as BR-GRU.

2.2. Multi-label dependency methods

In contrast to BR approaches there are multi-label dependency methods that are trying to utilize label dependency information. In the following, we provide a brief discussion of three important methods we will later use for our analysis together with ML-LSTM, ML-GRU and ML-CNN. It is important to note that these methods are known to give the most competitive results for the AAPD and Reuter-21578 datasets, which we will also use for our detailed comparison study. We will focus our scope of research on the methods themselves without any sophisticated pre/post-processing pipelines, hence, some alternative techniques for multi-label learning such as multi-label feature selection [45,46] and classifier chains [14] will remain to be explored in future studies.

- BERT with Balanced Loss (BERT-BL): Huang, Yi, et al. [47] proposed a distribution balancing loss (BL) function for applications in natural language processing. Such a loss function is especially suited for the learning of multi-label classification where the data have a long-tailed class frequency distribution. The reason therefor is that methods with such a loss function are able to address the label imbalance issue and in addition can utilize label dependency information to enhance the performance. In the following, we will refer to this model as BERT-BL.
- Label-Embedding Bi-directional Attentive Model (LBA): Liu, Naiyin, Qianlong Wang, and Jiangtao Ren proposed a novel architecture based on BERT [20]. LBA extends BERT by adding label-embeddings and a bi-directional attention mechanism to help BERT to learn relations from label dependency information on both the sequence- and token-level. They pointed out that BERT often struggles to fully harness token-level text representations and label embeddings. LBA is widely considered the first approach to exploit such information by using a BERT architecture. Importantly, their results showed a very competitive performance for several datasets, including the data from Reuters.
- Label correlation aware multi-label text classification (LACO): The LACO model by Zhang, Ximing, et al. [21] addresses the error propagation and overfitting problem from a sequence label prediction architecture. LACO extends BERT by introducing a joint embedding to obtain a text and label representation. Additionally, besides the native prediction layer, they also added a further label co-occurrence prediction layers. In terms of performance, numerical results demonstrated that LACO exhibits a highly competitive performance on the AAPD and RCV1-v2 datasets.

3. Methods

In this section, we describe the methods we use for our analysis. In the following, we distinguish between methods from two families, the LD (label dependency) family and the BR (binary relevance) family, which we define as follows.

Definition 3.1 (*LD Family*)**.** Methods from the LD family utilize explicitly label dependency information for MLTC.

Definition 3.2 (*BR Family*)**.** Methods from the BR family do not utilize label dependency information for MLTC.

Examples for methods from the LD family that we use for our analysis are BERT-BL, LBA and LACO and methods from the BR family we use are BR-SVM, BR-support, BR-decoder, BR-CNN and BR-LSTM and BR-GRU. All of these methods, except our fine-tuned BR-CNN, have been introduced previously. For this reason, we describe next the BR-CNN in detail.

3.1. Working mechanism of the BR-CNN

The convolutional neural network (CNN) structure we use is illustrated in Fig. 1, see [41]. From this structure one can see that the raw texts are first preprocessed to become tokens and then each token is mapped to a vector word embedding of 300 length. Let $x_i \in R^{300}$ denote the token at the ith position of the input sequence of total length l. The next layer is the convolutional layer where convolution filters $w \in R^{n\times 300}$ will be applied to the n-gram of the input sequence $x_{i:i+n-1}$ to extract abstract features at position i. Each convolution operation results in a feature map c_i given by

$$c_i = f(w * x_{i:i+n-1} + b). \tag{1}$$

Here, f denotes an activation function, b denotes the bias term, and $*$ is the convolution operation. With a step size of 1, the convolution operation will be applied to $x_{i:i+n-1}, i \in \{1, 2, 3, \dots, l\}$, resulting in a feature map

$$C = [c_1, c_2, c_3, \dots, c_{l-n+1}]. \tag{2}$$

After the convolutional layer, a max-pooling operation will be applied to each feature map C to extract the maximum value from the corresponding feature map. A vector collection of pooled values will be formed from all these feature maps

$$P = [\max(C_1), \max(C_2), \dots, \max(C_j)] \tag{3}$$

where j is the total the number of different filters. The final feature vector P will be fully connected to 2 output neurons to obtain the probability of binary classes $\hat{y}^0$ and $\hat{y}^1$.

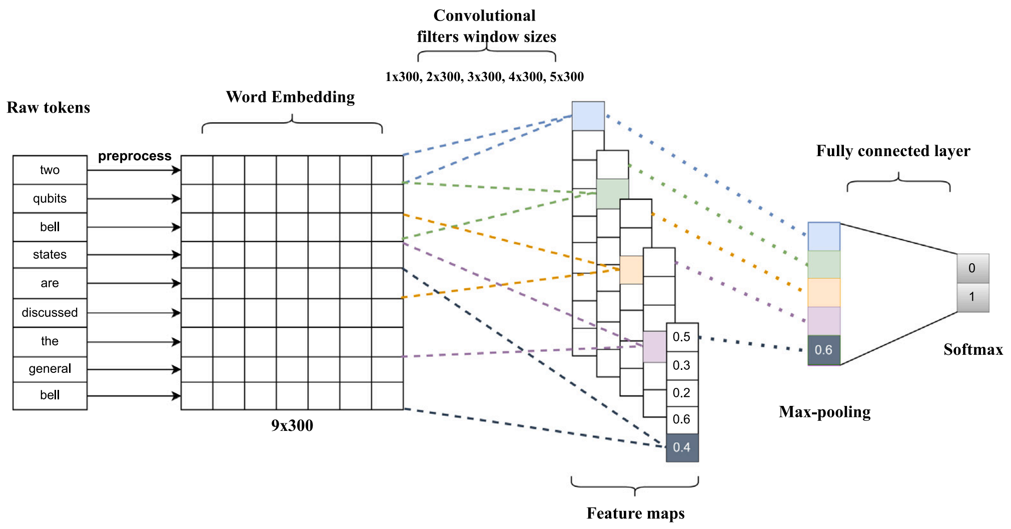

Fig. 1. The structure of the convolutional neural network (CNN) we use in our analysis as BR-CNN. This CNN consists of a word embedding layer, convolutional and pooling layers. The last layer of this network is fully connected.

3.1.1. *Weight scaling for BR-CNN*

In general, the loss function in a neural network is used to calculate the distance between the predicted label and the true label. Frequently, cross-entropy is used for this operation [48]. Then the calculated loss is used for the backward propagation to fine-tune the parameters of the network. However, a common problem is the imbalance of samples in the classes leading to overfitting. Usually, this problem can be explained by the contribution of the loss values from the more frequent class, which is in such a case much larger than the loss values from the other class. In this case, it may be desirable to adjust the loss values for both classes in accordance with certain weights. Properly scaling the loss can mitigate the influence of overly frequent classes and promote a more effective learning behavior.

For the scaling of the loss, we use the following weight parameters given by:

$$W_0 = \frac{1}{N_0} \quad , \quad W_1 = \frac{\alpha}{N_1} \tag{4}$$

In Eq. (4), N_0 corresponds to the total number of negative samples, N_1 is the number of positive samples and α is the scaling parameter. The resulting weights W_0 and W_1 correspond to the assigned weights of the negative samples and positive samples respectively. It is interesting to note that for our data usually $N_0 > N_1$ holds. For this reason, we use α values of 1, 10, and 100, which allow to increase the impact of positive samples by increasing the value of α and respectively of W_1.

As a consequence, the calculation of the weight-scaled binary cross entropy loss for sample k is given by:

$$L_k = -(W_0(y_k \cdot log(\hat{y}_k^0)) + W_1((1 - y_k) \cdot log(1 - \hat{y}_k^1))) \tag{5}$$

Here $k = \{1, 2, 3, \ldots, K\}$ where K is the total number of samples and y_k is the true label of sample k which is either 0 or 1. Furthermore, $\hat{y}_k^0$ is the predicted probability for sample k on class 0, and $\hat{y}_k^1$ is the predicted probability for sample k on class 1.

From this follows the final loss, L, given by

$$L = \frac{1}{K} \sum_{k=1}^{K} L_k. \tag{6}$$

Overall, L corresponds to the average value of L_k over all instances.

4. Data

For our numerical analysis, we use four datasets that have been previously used as benchmark data: AAPD, Reuters-21578, MIMIC-III and RCV1-v2. AAPD contains 54 different classes, Reuters-21578 contains 90, MIMIC-III contains 10 and RCV1-v2 contains 101 classes. An overview of the statistics of all 4 datasets can be seen in Table 1.

4.1. *Arxiv Academic Paper Dataset (AAPD)*

The AAPD data were proposed by Yang, Pengcheng, et al. [24]. This dataset consists of abstracts and the corresponding subjects of 55,840 papers collected from the field of computer science from the Arxiv website. In total, the abstracts are assigned to 54 different subjects corresponding to classes where each instance can be assigned to multiple classes (but having at least one label).

4.2. *Reuters-21578*

The Reuters data are a collection of Reuters newswire stories collected between the period from 1987 to 1991 [25]. The dataset contains more than 20,000 documents and each document is manually annotated into 90 unique categories corresponding to classes. The data have been also widely used as benchmark dataset for text classification [49]. For our analysis, we use the ApteMod version of Reuters-21578 which has 10,788 documents in total. For our analysis, we follow the training/testing splitting used by [50] which separates the data into 7769 training instances and 3019 testing instances.

4.3. *MIMIC-III*

The MIMIC-III (Medical Information Mart for Intensive Care) data [26] is a publicly available dataset providing a large variety of clinical information of patients. In our study, we used the annotated discharge summaries from the work [26] where up to 1610 discharge summaries were annotated into 10 different categories. Given the discharge summaries, the task is to predict the corresponding phenotypes where each discharge summary (corresponding to a patient) can be assigned to more than one phenotype. For this reason, also, phenotype categorization represents a multi-label text classification problem.

Table 1
Overview of the data characteristics for the four datasets from AAPD, Reuters-21578, MIMIC-III and RCV1-v2 we use for our analysis. Number of frequent classes correspond to the number of classes that have at least 5% of true labels of the total training samples, while number of rare classes indicate the labels with less than 0.1%.

	AAPD	Reuters-21578	MIMIC-III	RCV1-v2
Training size	5,3840	7769	1207	23,149
Testing size	1000	3019	403	781,265
Average tokens per sample	153	80	2067	122
Number of classes	54	90	10	101
Average number of labels per sample	2.40	1.23	1.56	3.24
Number of frequent classes (>5%)	17	5	10	16
Number of rare classes (<0.1%)	0	26	0	10

4.4. RCV1-v2

The Reuters Corpus Volume I (RCV1-v2) data [27] is a another commonly used benchmark dataset for multi-label text classification. It is composed by more than 800,000 newswire stories and each story is manually annotate into 103 different classes (101 in the training set). RCV1-v2 is an extended version of Reuters-21578. We followed the same training/testing partitioning of the data from the original work in [27].

5. Results

In this section, we present our numerical results. First, we study the optimization of the BR methods in details. Second, we conduct a comparison between all MLTC models. Third, we investigate learning curves of the best performing BR method and LACO. Fourth, we study the impact of the number of classes on the performance. Fifth, we conduct an ablation study for the best performing BR method.

5.1. Optimization of BR-CNN

In this subsection, we study the optimization of BR models. Specifically, we study the tuning of the parameters of the BR-CNN. In order to obtain the best performing BR-CNN, one needs to optimize its structure and parameters. This includes the fine-tuning of filter window sizes, the number of windows per window size, the embedding sizes and the weight scaling of the loss function. Based on this optimization process, we select the combination of parameters that provides the best results.

We perform fine-tuning for all classes, although one of the advantages of employing binary relevance learning in multi-label learning problems is the ability to apply the fine-tuning process to each individual class. However, such an approach would be very time consuming. When discussing the results from a comparative analysis in Section 5.2 this approach will be substantiated by the results.

In Fig. 2, we show the results of the optimization process of BR-CNN for the AAPD data. The vanilla BR-CNN uses a 1×1 filter window size, 100 filter windows per window size and an embedding size of 50 without weight scaling. We start the optimization process firstly by adding larger sizes of filter windows as in Fig. 2(a). Here we observe a stable performance after adding a filter window size of 1×5. Then we increase the number of filter windows per window size, as in Fig. 2(b). From this we see that 300 filter windows per window size gives the best performing model. Thereafter we change the size of the embedding from 50 to 300. As one can see from Fig. 2(c), we find the best performing model using an embedding size of 300.

The optimization process for the Reuters-21578 data follows a similar pattern. Specifically, when optimizing parameters such as the filter window size, the number of filter windows, the embedding size, and the weight scale, we obtain very comparable results to those observed for the AAPD data, as shown in Fig. 2(a) to (c). A parallel optimization pattern is evident for the MIMIC-III and RCV1-v2 datasets, with the

Table 2
Parameters settings for BR-LSTM and BR-GRU as used for our analysis.

Hyperparameters	BR-LSTM	BR-GRU
Number of layers	1	1
Number of neurons	512	512
Bidirectional	Yes	Yes
Optimizer	Adam	Adam
Embedding	Same as BR-CNN	Same as BR-CNN

Table 3
Results for a comparison between multi-label deep learning methods (top) and BR methods (middle) for micro- and macro-F1-scores. Bottom: Summary of the performance difference between top performing BR and the top performing (SOTA) label dependency utilizing method on AAPD and Reuters-21578.

Model	AAPD		Reuters-21578	
	Micro	Macro	Micro	Macro
BERT-BL [47]	0.721	0.565	0.906	0.647
LBA [20]	0.721	0.571	**0.908**	**0.675**
LACO [21]	0.749	0.612	0.880	0.600
ML-LSTM [42]	0.660	0.478	0.815	0.350
ML-GRU [43]	0.670	0.495	0.850	0.439
ML-CNN [41]	0.715	0.520	0.852	0.470
BR-SVM [1]	0.646	0.529	0.851	0.402
BR-support [20]	0.696	0.545	0.855	0.440
BR-decoder [19]	0.700	0.521	0.861	0.446
BR-LSTM [42]	0.680	0.501	0.830	0.390
BR-GRU [43]	0.720	0.563	0.870	0.465
BR-CNN	**0.750**	**0.645**	0.891	0.558
ΔF(BR/SOTA)	(+0.1%)	(+5.1%)	(−1.8%)	(−17%)

exception of embedding tuning for MIMIC-III. In the case of MIMIC-III, a fixed dimension of 50 is used, based on word2vec pre-trained on all the summaries from the MIMIC-III dataset.

Consequently, for the forthcoming analysis employing BR-CNN, we will use filter window sizes of 1×1, 1×2, 1×3, 1×4, and 1×5, each having 300 filter windows. We will apply a dropout rate of 50% after max-pooling and utilize Glove-6B embeddings of length 300 for AAPD, Reuters-21578, RCV1-v2, and word2vec embeddings of length 50 for MIMIC-III. In all models, a weight scaling of $\alpha = 1$ will be utilized. Additionally, during AAPD testing, we observed that models trained with two different sets of parameters yield the best results for macro-F1. Specifically, these two sets involve 300 and 400 filter windows per window size, respectively, while keeping all other parameters constant. We will select models with the highest F1 score for each class from these sets and employ them to calculate the final micro-F1 and macro-F1 scores.

A similar optimization process is conducted for both BR-LSTM and BR-GRU each using one recurrent layer followed by two fully connected layers of 256 neurons. The optimal hyperparameter configuration of BR-LSTM and BR-GRU is shown in Table 2.

5.2. Comparison of methods

Using the optimized parameters of the BR-CNN, BR-LSTM and BR-GRU for the AAPD, Reuters-21578, MIMIC-III and RCV1-v2 data, we can now compare the deep learning BR models with other methods from the literature. We summarizes the results of this analysis showing the performance of the methods in comparison with the top performing MLTC methods utilizing label dependency information (BERT-BL, LBA, LACO) and baseline BR methods (BR-SVM, BR-support and BR-decoder). Furthermore, we study ML-CNN, ML-LSTM and ML-GRU which are the algorithm transformed versions of the corresponding deep learning BR methods. Table 3 shows the results for AAPD and Reuters-21578 and Table 4 shows the results for MIMIC-III and RCV1-v2.

First, we would like to remark that the BR methods are quite heterogeneous. That means BR-SVM and BR-support are both based on

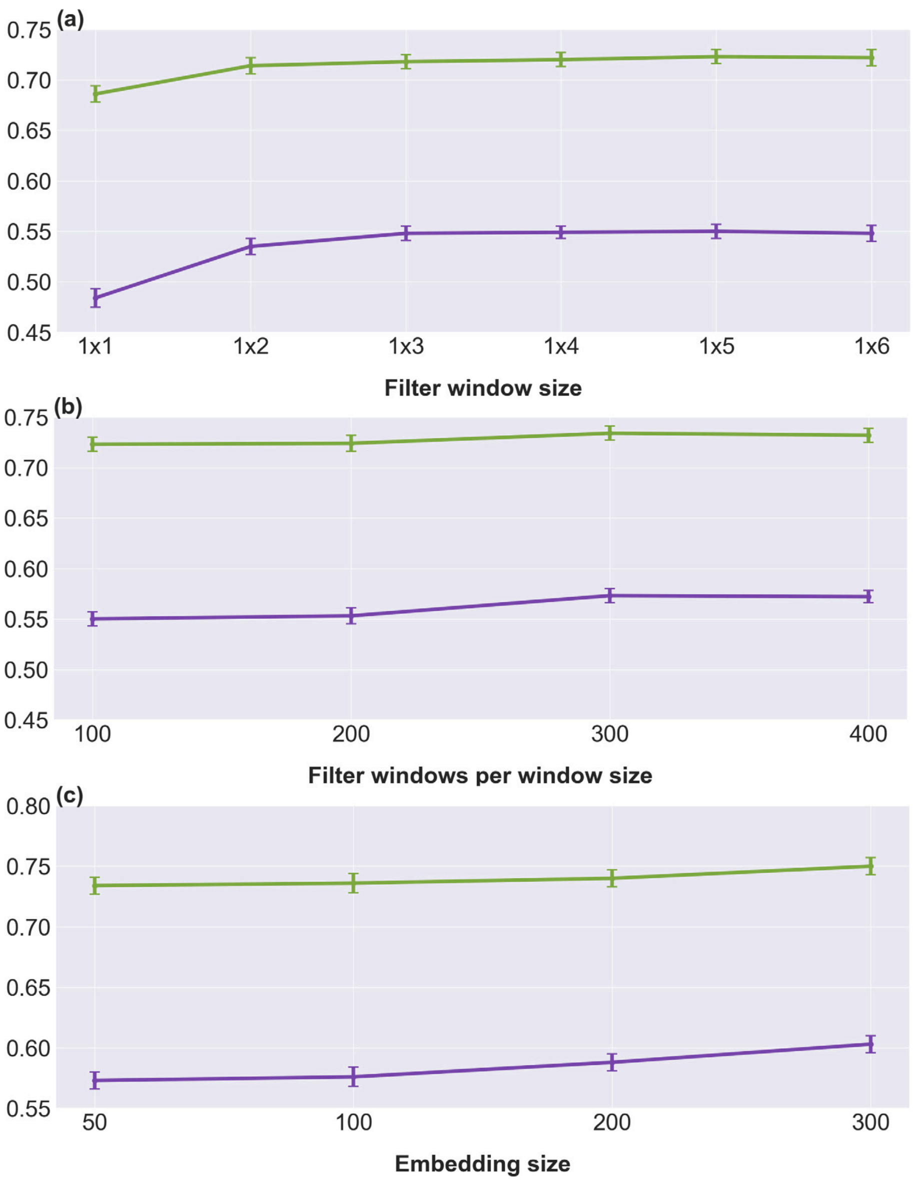

Fig. 2. Optimization of hyperparameter settings of the BR-CNN for AAPD data. (a) Impact of the filter window size. (b) Impact of the number of windows per window size added based on the best performance from subfigure (a). (c) Impact of the embedding size based on the best performing model from (b).

Table 4
Results for a comparison between multi-label deep learning methods (top) and BR methods (middle) for micro- and macro-F1-scores. Bottom:Summary of the performance difference between top performing BR and the top performing (SOTA) label dependency utilizing method on MIMIC-III and RCV1-v2.

Model	MIMIC-III		RCV1-v2	
	Micro	Macro	Micro	Macro
BERT-BL [47]	0.620	0.623	0.880	0.701
LACO [21]	0.635	0.645	**0.885**	**0.731**
ML-LSTM [42]	0.260	0.200	0.815	0.590
ML-GRU [43]	0.330	0.315	0.824	0.618
ML-CNN [41]	0.636	0.632	0.854	0.670
BR-SVM [1]	0.458	0.452	0.852	0.662
BR-decoder [19]	0.412	0.405	0.867	0.671
BR-LSTM [42]	0.320	0.290	0.830	0.642
BR-GRU [43]	0.335	0.320	0.845	0.660
BR-CNN	**0.725**	**0.728**	0.872	0.692
ΔF(BR/SOTA)	(+12.4%)	(+11.4%)	(−1.4%)	(−5.3%)

traditional machine learning classifiers whereas BR-decoder, BR-CNN, BR-LSTM and BR-GRU utilize deep learning. Regarding the results, from Table 3 one can see that the results for the AAPD and Reuters-21578 data are different from each other. While for Reuters-21578, LBA is overall best with a slight advantage over BERT-BL, for the AAPD data BR-CNN is the top performer followed by LACO. However, for both datasets, all the BR methods expect for BR-CNN are outperformed by the top performing models LBA, LACO and BERT-BL. Looking at the margins, we find for the AAPD data regarding the micro-F1 and macro-F1 that BR-CNN is better by 0.13% and 5.10% respectively compared to LACO. For the Reuters-21578 data, all of the BR methods are inferior to LBA especially on macro-F1 while LBA outperforms BR-CNN by 17.0%. The last line in Table 3 summarizes the performance difference between the top performing BR and top performing (SOTA) label dependency utilizing methods of the corresponding category.

For MIMIC-III and RCV1-v2, the results can be seen in Table 3. We find that for MIMIC-III, BR-CNN tops the performance and outperforms LACO by a large margin of 12.4% and 11.4% on micro-F1 and macro-F1 respectively. As for RCV1-v2, the distances between F-scores for all methods are the smallest within all the 4 datasets. LACO has the best

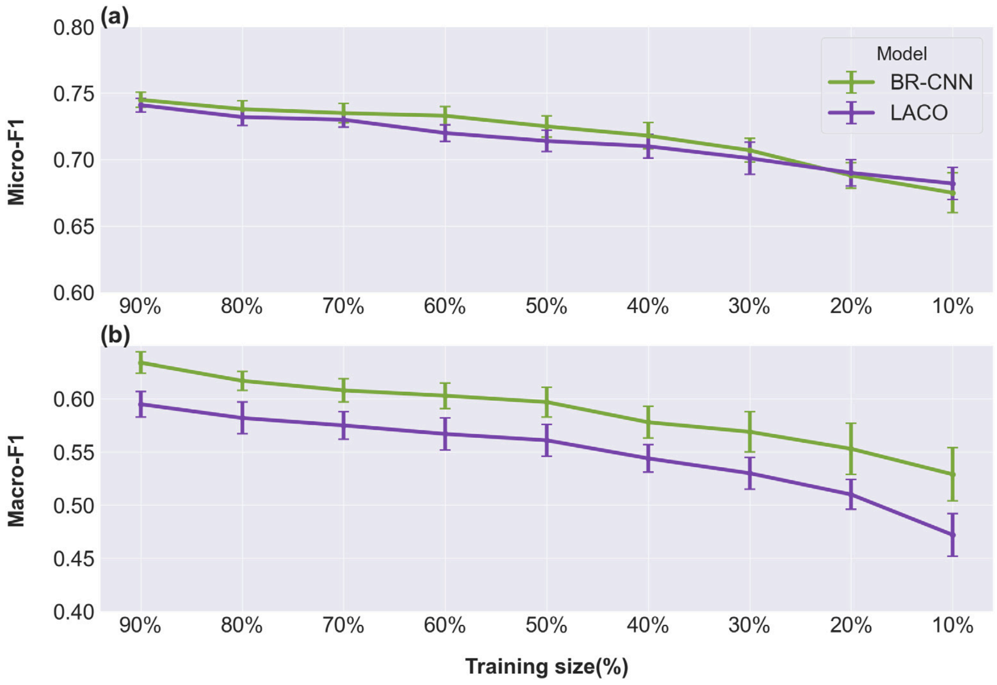

Fig. 3. Learning curves for BR-CNN and LACO for the AAPD data. Performance in dependence on the size of the training data. (a) Micro-F1. (b) Macro-F1.

performance on both micro- and macro-F1, which outperforms the top BR method BR-CNN by 1.4% and 5.3% respectively. The last line in Table 4 summarizes again the performance difference between BR-CNN and the top performing (SOTA) label dependency utilizing method of the corresponding category.

Overall, for all the methods in the BR family, it is notable that BR-CNN is the best performing BR method across all 4 datasets. Furthermore, BR-CNN has also the top micro- and macro-F1 among all the tested methods on the AAPD and MIMIC-III datasets. Another interesting observation is that BR-GRU constantly outperforms BR-LSTM on the 4 datasets, even though BR-GRU has lesser parameters than BR-LSTM and both have a similar structure.

In order to obtain insights into the computational complexity of the top performing methods, we report the runtime of training BR-CNN and the multi-label learning models. All the neural network models were trained using a 4-way NV-LINK configuration of the Tesla V100. For the AAPD data, training LACO with 40 epochs took approximately 2 h, while training BR-CNN with 25 epochs and early stopping which stop training if F1 score is not improving in 5 epoch, required approximately 1.6 h. For the Reuters-21578 data, training BERT-BL with 40 epochs took approximately 0.6 h, and training BR-CNN with 20 epochs and early stopping required approximately 2.3 h. In the case of MIMIC-III, training BR-CNN took approximately 0.5 h with 15 epochs. Additionally, LACO with 20 epochs took approximately 0.4 h, and BERT-BL with 20 epochs took 0.5 h. For the training of RCV1-v2, BERT-BL with 30 epochs took approximately 5.4 h and BR-CNN with 15 epochs took approximately 12 h.

5.3. Learning curves

In order to examine the impact of the size of the training data on the performance of the best performing algorithms, as found in the previous section, we study the learning curves for BR-CNN and LACO. For this analysis, the size of the training of the AAPD data is successively reduced from 90%, 80%... to 10%. We train LACO and BR-CNN for each subset and evaluate them on the same test set to learn how the size of the training set affects the performance of both methods.

Fig. 3 shows the results for the learning curves. For decreasing sizes of the training data there is a steady but moderate decrease in the performance for both BR-CNN and LACO. Importantly, increasing training sizes beyond 50% leads only to a very modest increase in performance indicating that the total training data are sufficient to reliably training both models. Furthermore, it is interesting to note that for macro-F1 the BR-CNN is consistently better than LACO. In contrast, for micro-F1 the top performing model changes but BR-CNN and LACO are very similar across the different sizes of the training data.

5.4. Impact of the number of classes on the performance

For the next analysis, we aim to investigate the impact of the number of classes on the performance. Since there is no single unique procedure for selecting subsets of classes, we will examine three distinct selection mechanisms to obtain these subsets of classes. Specifically, we will study subsets from random sampling, the ranking of co-occurrence levels and the ranking of class frequencies.

Random sampling: First, we study the selection of classes based on random sampling. In order to obtain robust estimates, we average over 5 independent runs. The results of this are shown in Fig. 4. The x-axis shows the number of classes randomly sampled from all available 54 classes. As one can see, the mean values for both macro-F1 and micro-F1 are quite stable for BR-CNN and LACO. However, the performance of LACO is more effected by the number of classes than BR-CNN. Furthermore, the standard error is clearly increasing for smaller subsets. This is reasonable because the variation becomes larger when fewer classes are sampled. This variation points also to a heterogeneity of the individual classes. We will come back to this issue when ranking the classes according to the co-occurrence levels and class frequencies.

Ranking of co-occurrence levels: Next, we repeat such an analysis, but for the selection of classes based on co-occurrence levels. This will allow to obtain a ranking of classes. We define the co-occurrence level in the following way: For the AAPD data, we have 54 classes. Let us call the (symmetric) co-occurrence matrix J with $J_{i,j} \in \mathbb{N}$, with $i = \{1, \dots, 54\}, j = \{1, \dots, 54\}$. Here $J_{i,j}$ indicates the number of instances for which one can find label i together with label j. Then, we divide

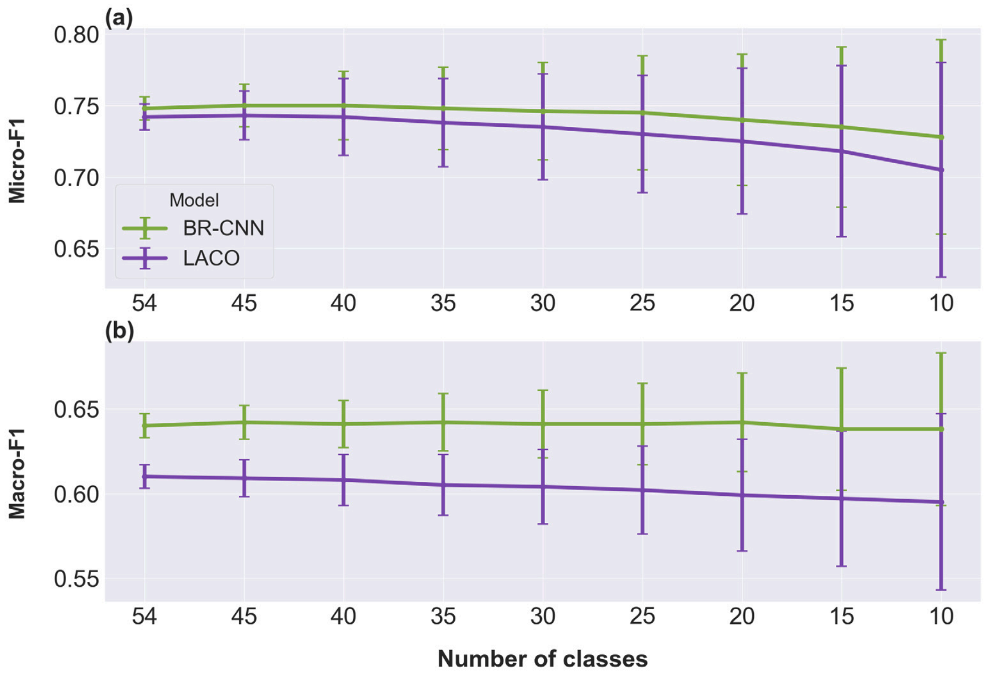

Fig. 4. Random sampling of subsets of classes. Performance of BR-CNN and LACO for the AAPD data. (a) Micro-F1. (b) Macro-F1. The x-axis indicates the number of classes obtained from random sampling.

each row of the matrix J by its maximum value, i.e., $\tilde{J}_i = J_i/max(J_i)$ where J_i corresponds to the ith row of matrix J and $\tilde{J}_i$ is a standardized vector. This gives a new matrix $K = (\tilde{J}_i)_{i=1}^{54}$ with standardized rows. From this we take the row summation given by

$$C_i = \sum_{j}^{54} K_{i,j}. \tag{7}$$

This is the final vector containing co-occurrence levels C_i for each class i.

Fig. 5(a) shows the co-occurrences levels C_i for all 54 classes of the AAPD data. One can see that there are large differences between the classes especially for small and large values of C_i at the beginning and end of the distribution.

For constructing subsets of classes, we start by selecting classes with the smallest co-occurrence levels first. This way the ranking of the co-occurrence levels is systematically utilized for obtaining subsets of a certain size. The results for BR-CNN and LACO are shown in Fig. 5(b,c). In contrast to the results from the random sampling, we observe now a nonlinear dependency of the macro-F1 and micro-F1 on the size of the subsets. However, the degree of nonlinearity is different for BR-CNN and LACO. Specifically, for BR-CNN we find the maximum value of macro-F1 and micro-F1 for the smallest subset, i.e., for 5 classes. Also the maximum macro-F1 value for LACO is for 5 classes, however, the maximum micro-F1 is observed for 35 classes.

Ranking of class frequencies: The last selection mechanism we study is based on the ranking of class frequencies. That means, for this selection, we rank order the classes according to their sample frequencies belonging to a particular class. In Fig. 6(a), we show this ranking for the AAPD data. For each class, the number of samples with the corresponding label is shown. Based on this ranking, we study two different types of subsets. For the first, we select subsets starting on the right-hand-side of the class frequency distribution and for the second, we start on the left-hand-side. That means, the first selects the smallest classes whereas the second selects the largest classes.

The result of this analysis are shown in Fig. 6(b), and (c). For all of these subsets, BR-CNN performs better than LACO and the differences are typically larger than the standard error. An interesting observation is that the number of classes alone does not provide a conclusive indication of performance. Specifically, when considering the ascending order of the classes (left side), the best performance of BR-CNN and LACO is achieved with the largest number of classes (which is 20). However, in the case of the descending order (right side), the best performance is achieved with the smallest number of classes (which is 5).

We repeat this analysis for the Reuters-21578 data and the results are shown in Fig. 7. The size of the subsets of classes corresponds to 30, 50, 60, 70 and 80 for Fig. 7(b) (class ranking in ascending order) and 5, 15, 40, 60 and 80 for Fig. 7(c) (class ranking in descending order). In Fig. 7(b) one can again observe a nonlinear behavior for BR-CNN and BERT-BL. Interestingly, for Fig. 7(c), we see a different behavior than for Fig. 7(b). Here BR-CNN performs better for smaller subsets of classes while for larger subsets BERT-BL is better. The distance between both models is even larger than the standard error.

Class-wise F1-scores: To complement our preceding analysis, we study finally the individual (class-wise) F1-scores for AAPD and Reuters -21578 for the best performing models in Table 3. The results are shown in Fig. 8. Here the F1-score for each class is reported for BR-CNN and LACO for AAPD (see Fig. 8(a)), and BR-CNN and BERT-BL for Reuters-21578 (see Fig. 8(b)). As one can see from the figures, the models' behaviors for AAPD are more stable, while for the Reuters-21578 data, we observe several classes with F1-scores of zero. This indicates a severe heterogeneity within the data. We will come back to this point in the next section.

5.5. Ablation study

Finally, we study the independent effect of important components of our BR-CNN to see how this affects its performance. This corresponds to an ablation study. Originally, ablation studies have been introduced in neurobiology to gain a better understanding of the functioning of the brain [51,52]. Basically, an ablation study involves removing a specific part of the brain to observe any resulting changes on its functional behaviors. Recently, a similar approach has been suggested to investigate the working behavior of deep learning architectures [53,54]. In this section, we perform an ablation study for our BR-CNN.

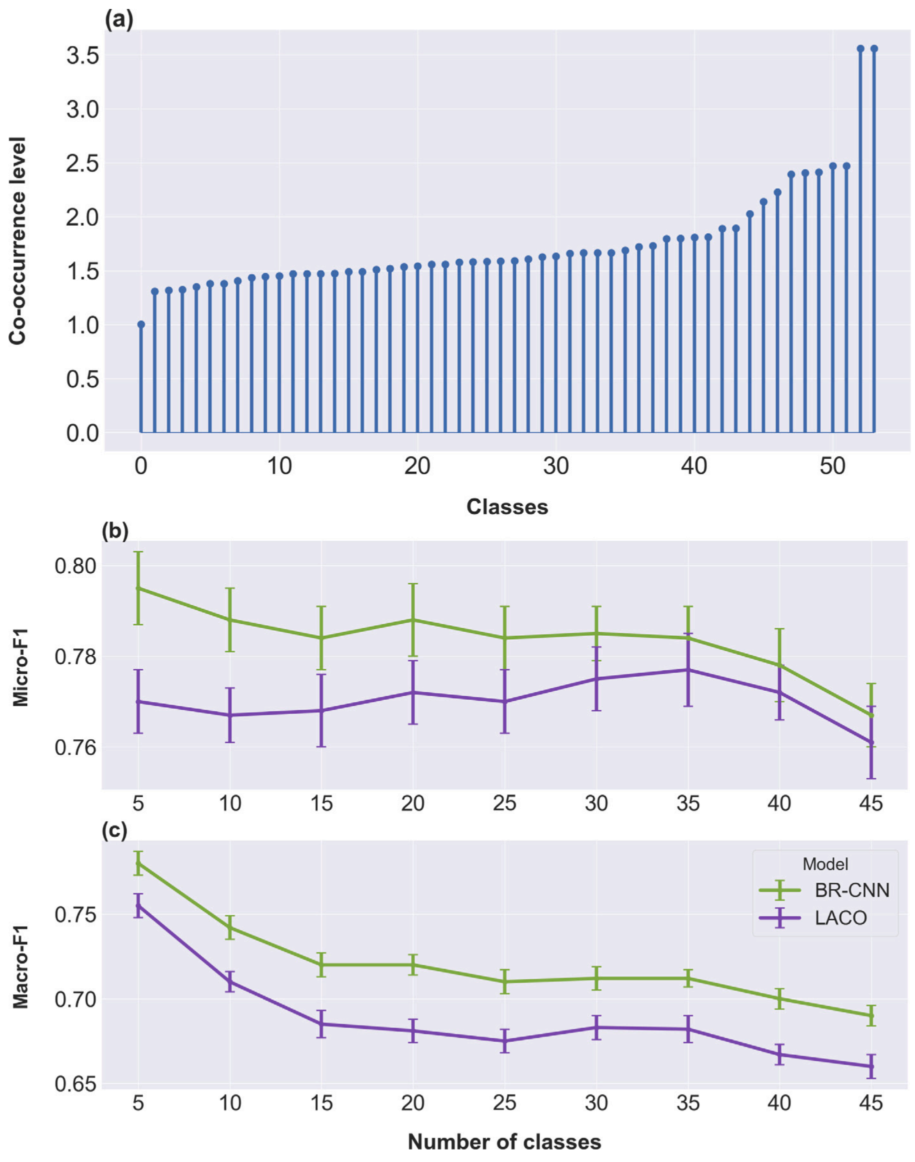

Fig. 5. (a) Rank ordered co-occurrence levels for all 54 classes of the AAPD data based on values of the score C_i (see Eqn. 7). (b) and (c) show results for subsets of classes based on rank ordered co-occurrence levels given in (a). Performance of BR-CNN and LACO in dependence on the number of classes for AAPD data. The x-axis indicates the number of classes.

Impact from removing components: We start by investigating the effect of weight scaling we use in our BR-CNN. For this, we study BR-CNN for all 4 datasets with and without application of a weight scaling function. The results for the macro-F1 are shown in Table 5. As one can see, with a weight scaling (first row) one can improve the macro-F1 on BR-CNN for all the datasets, whereas without weight scaling (second row) we see the most significant performance loss for the Reuters-21578 dataset with 20.2%, followed by 18.6% for RCV1-v2. For the other 2 datasets, which are having more balanced label frequency distributions, the improvements are not significant with weight scaling. This demonstrates numerically that weight scaling enhances the learning on rare classes.

The next component we study is the pre-trained word embedding vectors. For this, we remove the pre-trained word embeddings and use only the randomly initialized word embeddings for our training. We would like to note that all the parameters for the embedding are still adjustable. Overall, all methods show a decrease in performance (see third column) when the word embedding vectors are random initialized. However, the amount of performance change depends on the dataset.

Table 5
Effect of an ablation study on the performance by modifying different components of a BR-CNN. The shown numbers correspond to the macro-F1. Second row: Results without weight scaling. Third row: Results without pre-train word vector.

	AAPD	Reuters-21578	MIMIC-III	RCV1-v2
BR-CNN	**0.645**	**0.558**	**0.725**	**0.692**
- w/o weight scaling	↓ 0.620	↓ 0.445	↓ 0.718	↓ 0.563
- w/o pre-train word vector	↓ 0.615	↓ 0.504	↓ 0.680	↓ 0.645

Impact from the weight scaling parameter: The effects of the weight scale function can be adjusted by the parameter α; see Eq. (4). This allows us to investigate the effect of 3 different values of α, given by 1, 10 and 100, to see its impact on macro-F1. The results of this analysis can be found in Table 6. From this we discover a pattern for all datasets that a value of $\alpha = 1$ provides the best results for BR-CNN for all datasets. Further increasing the value of α leads to a lowering of the performance. On the other hand, it is interesting to note that for Reuters-21578 and RCV1-v2 using α values larger than 1 leads to an improvement in the performance over the BR-CNN without weight scaling (first row in Table 6).

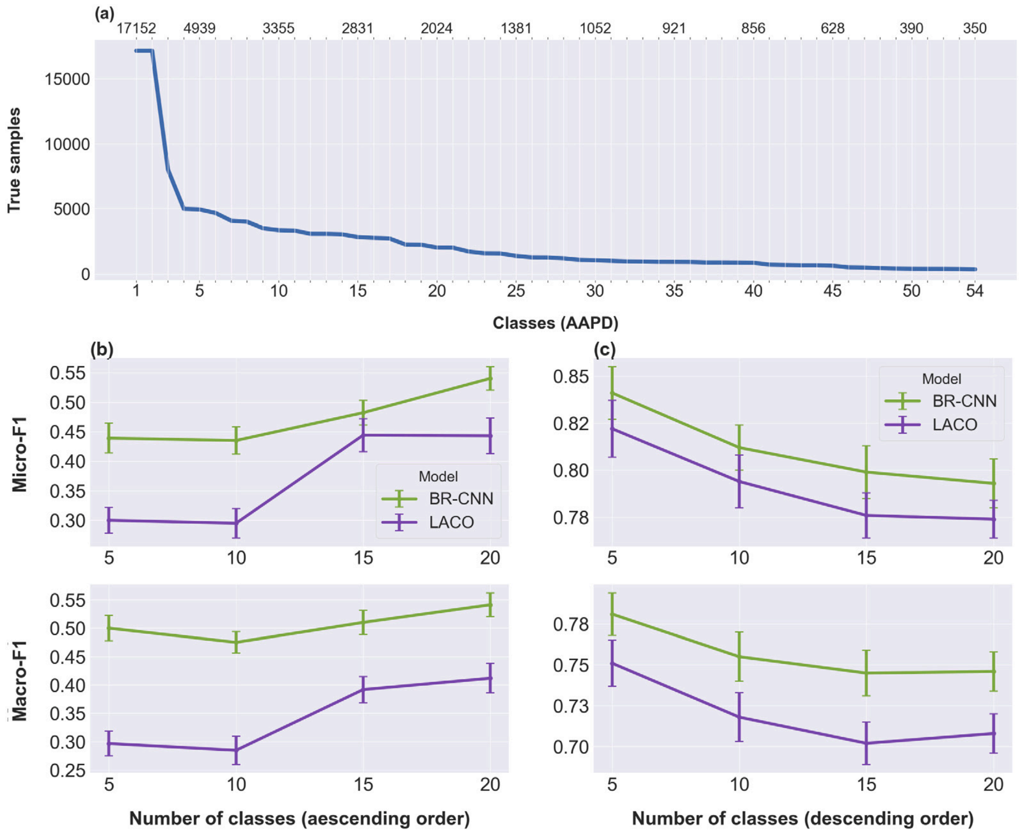

Fig. 6. (a) Rank ordered distributions of class frequencies for the AAPD data. The y-axis gives the number of samples having a certain class label. The specific number for every 5 classes is shown on the top of each figure. (b)–(c): Performance comparison between LACO and BR-CNN for subsets of classes based on the rank ordering of classes according to the frequency of samples. Column (b): Results for the least frequent classes. Column (c): Results for the most frequent classes.

Table 6
Effect of different α values on the weight scaling function. The shown numbers correspond to the macro-F1.

BR-CNN	AAPD	Reuters-21578	MIMIC-III	RCV1-v2
No weight scaling	0.620	0.445	0.718	0.563
$\alpha = 1$	↑ **0.645**	↑ **0.558**	↑ **0.725**	↑ **0.692**
$\alpha = 10$	↓ 0.615	↑ 0.542	↓ 0.665	↑ 0.670
$\alpha = 100$	↓ 0.575	↑ 0.520	↓ 0.420	↑ 0.615

6. Discussion

Multi-label text classification (MLTC) is an intricate task because the problem can be presented in a number of different ways. First, one can distinguish between multi-label and binary relevance classifiers where the former models are genuine multi-label classifiers while the latter transform the task into multiple binary classifications. Based on this, one can further distinguish between approaches that utilize deep learning or label dependency information. This categorization is shown in Fig. 9 including some methods. We would like to point out that in our analysis, we did not use a classifier chain [14] because studies found their inferiority to all the methods in Table 3, except BR-SVM [21,55].

In this paper, we explore a specific subset of Multi-Label Text Classification (MLTC) methods, focusing on Binary Relevance (BR) approaches within the realm of deep learning that do not rely on label dependency information. This constitutes a specialized area because within the MLTC community, there is a prevailing belief that leveraging label dependency information is essential for achieving optimal results. Furthermore, in recent years, a majority of these methods have been rooted in deep learning techniques. Specific examples for such approaches are BERT-BL [47], LBA [20] and LACO [21].

From our initial analyses using deep learning BR methods, we found that an off-the-shelf version of BR-CNN, BR-LSTM and BR-GRU does not give good performances. However, after careful optimization of the many hyperparameters of the models (see, e.g., Fig. 2), we noticed a considerable improvement of the models, especially for BR-CNN, after applying a weight scaling function. In order to obtain more insights into the performance of our BR-CNN, we conducted a comprehensive comparison with the best performing methods based on BR and label dependency information from the literature.

From this comparison (see Tables 3 and 4) we make a number of interesting observations. First, BR-CNN is the overall best performing classifier for the AAPD and MIMIC-III data. This observation underscores the fact that Binary Relevance (BR) not only competes favorably with the best-established methods in the literature but can also surpass them in terms of performance when applied to targeted datasets. This is also demonstrated for various subsets of the AAPD and Reuters-21578 data from our analysis (see, e.g., Figs. 4, 5 and 7). Second, for all studied data (not only the ones shown in Tables 3 and 4) BR-CNN is better than all other BR-based classifiers. Here it is important to note that also BR-decoder, BR-LSTM and BR-GRU are based on deep learning, hence, this is not the singular difference. Instead, in our opinion it is the optimization of all components of the neural network we use for our analysis that makes the difference. Third, the binary relevance transformed versions of deep learning methods (BR-CNN, BR-LSTM, BR-GRU) always outperform the algorithm adaptation transformed versions (ML-CNN, ML-LSTM, ML-GRU) on all the datasets we tested.

Considering the importance of the data on the results, we perform a number of additional comparisons for studying this in more detail. Specifically, when looking at the characteristics of the AAPD and Reuters-21578 data one can see a heterogeneity within the data, e.g., by

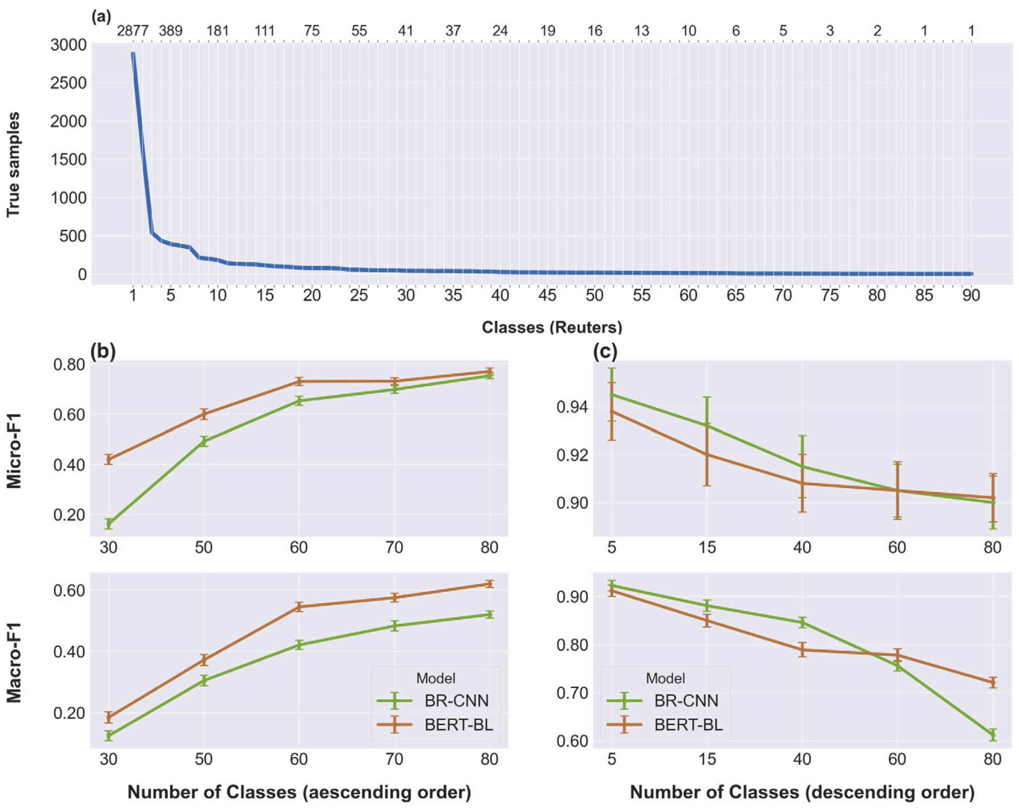

Fig. 7. (a) Rank ordered distributions of class frequencies for the Reuters-21578 data. The y-axis gives the number of samples having a certain class label. The specific number for every 5 classes is shown on the top of each figure. (b)–(c): Performance comparison between BERT-BL and BR-CNN for subsets of classes based on the rank ordering of classes according to the frequency of samples. Column (b): Results for the least frequent classes. Column (c): Results for the most frequent classes.

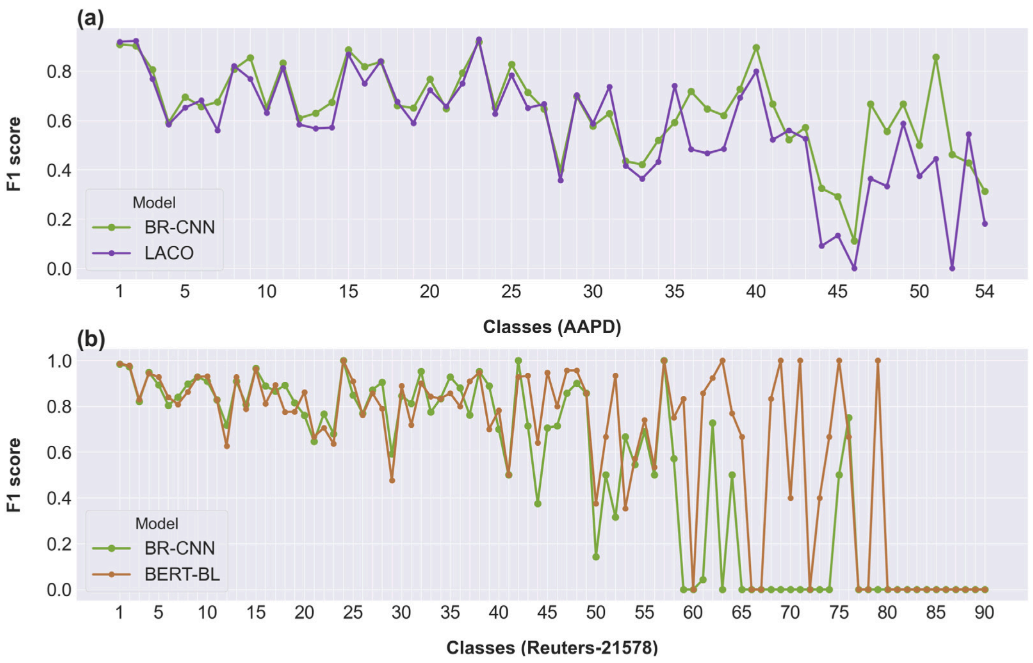

Fig. 8. Individual (class-wise) F1-scores for BR-CNN and LACO for AAPD (a) data and BR-CNN and BERT-BL for Reuters-21578 (b) data. The number of positive samples for each class on the x-axis are the same as in Fig. 6(a) for AAPD and Fig. 7(a) for Reuters-21578.

ranking co-occurrence levels (see Fig. 5) and class frequencies (see Figs. 6 and 7). For the class frequencies it is interesting to note that both datasets have long tail distributions while Reuters-21578 has an even longer tail than AAPD. Regarding the performance of the models, we find a nonlinear dependency on the number of classes (for BR-CNN and LACO) meaning that by increasing the number of classes the learning task is considerably more complicated putting potentially a limit on the total number of classes that can be learned.

We also found that for both datasets where BR-CNN outperforms SOTA methods, the number of frequent classes is essential to make such

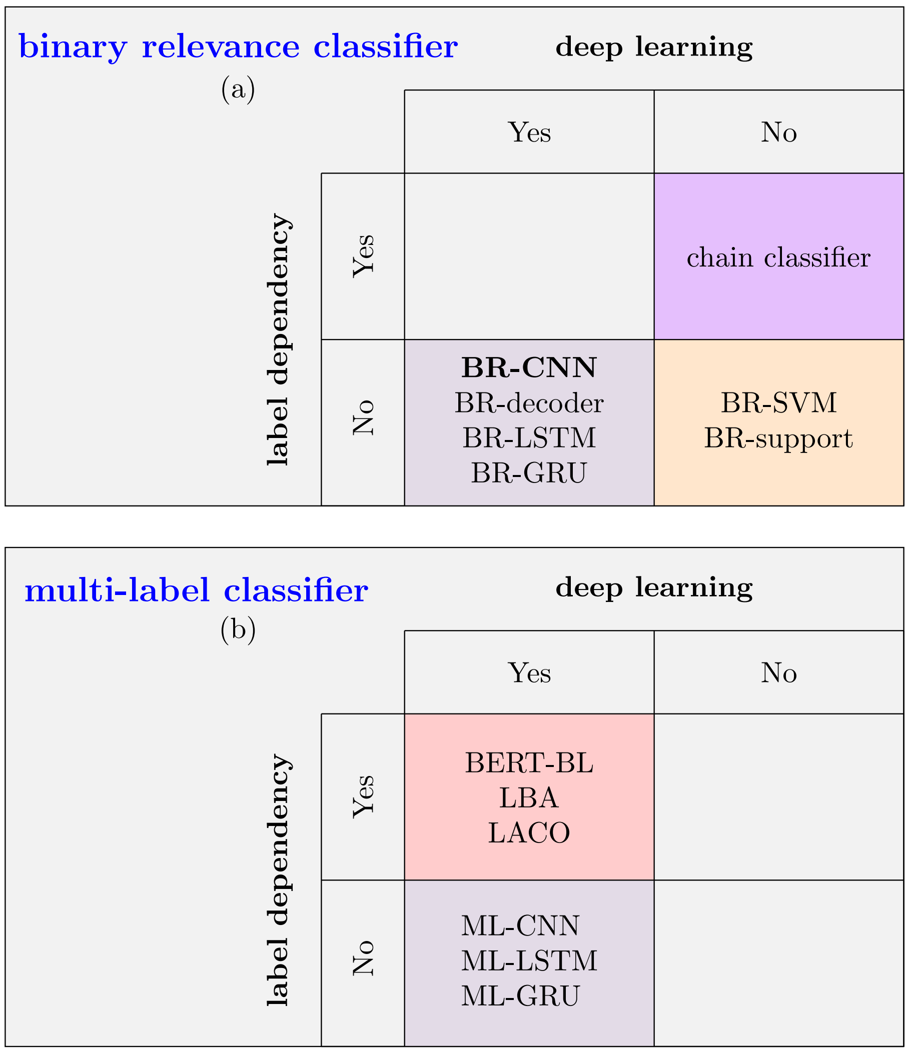

Fig. 9. Overview of combinations to categorize MLTC methods. Top table: Binary relevance-based approaches. Bottom table: Multi-label-based approaches.

a difference. According to Table 1, for MIMIC-III dataset, where all the classes are frequently present during training, we observe the largest performance gap between BR-CNN and novel architectures. For AAPD, where no rare classes are present, BR-CNN also outperforms all novel models but with a limited margin. For Reuters-21578 and RCV1-v2, both datasets contain a portion of rare classes – some of the classes even have under 4 training samples – one can observe inferior BR-CNN in comparison to other methods.

One of the most interesting results, we find when studying the number of classes added in ascending order according to the class frequencies (see Fig. 6 b). Due to the fact that the first classes considered are more difficult to learn (having fewer samples) an increase in the number of classes leads to an improved performance. Importantly, this is not a property of the BR-CNN but can be also observed for LACO. Of similar interest is the fact that for smaller subsets of classes for the Reuters-21578 data, BR-CNN can perform better than BERT-BL (see Fig. 7 c)). This is important because BERT-BL is the better performing model over all classes compared to BR-CNN (see Table Table 3). Both findings demonstrate that general statements about methods cannot be made without reference to the characteristics of the data. Overall, based on our findings, we would like to argue that BR-CNN can perform as well as the most competitive methods on targeted datasets.

Aside from these numerical findings, our study contributes to a more fundamental discussion related to multi-label classification. Specifically, according to [56] it seems *opinio communis*, which means the generally accepted view or consensus, that an optimal prediction performance for multi-label classification can only be achieved by methods that explicitly take the dependencies between the labels of the classes into consideration. Similar sentiments can be found widely, e.g., in [57–59]. In summary, this consensus implies that for any given task, one can always find a method from the LD (label dependency) family that consistently outperforms any method from the BR (binary relevance) family.

It is important to note, however, that while this prevailing view exists, it has not been formally proven. Furthermore, it is worth highlighting that despite the perceived dominance of LD methods, there are studies, such as Read et al. [14], cautioning against dismissing the effectiveness of BR methods.

Our study challenges the prevailing consensus regarding the superiority of LD (label dependency) methods, as evident from our results. However, it is essential to provide context and perspective for this. First, our focus was primarily on MLTC tasks, limiting the scope of our study to text data. Consequently, we cannot ascertain if our results extend to other data types such as image, audio or sensory data. A problem with a comparison of methods among different data types is that one cannot just apply the same analysis method to text, image or audio data but each data type requires usually a non-trivial adaptation of the analysis method. Hence, a comparison among data types is not straight forward at all and it remains to be seen if different data types favor different approaches.

Second, our findings indicate that the results are influenced by the characteristics of the text data, suggesting that no single method universally outperforms others across all studied data. This implies that text data from different domains, with distinct properties, may yield different outcomes when applying different methods. From this one can see that there is not only a difference between different data types but also within the domain of text data itself. This heterogeneity is certainly a reason for the observed complexity of MLC in general.

Third, until today the most effective approach to incorporating label dependency information remains uncertain. For this reason, label dependency information cannot "just" be added but this is a separate research question that requires a dedicated analysis. Therefore, it is possible that future methods within the LD family may achieve improved results, given advancements in methodology and techniques.

In summary, while our study challenges the prevailing consensus and highlights certain limitations, it is crucial to consider the domain-specific nature of our results and acknowledge the influence of the characteristics of datasets.

7. Conclusion

Multi-label text classification (MLTC) presents a challenging task that expands beyond binary classification. In general, such approaches can be classified into two distinct model families: the label dependency (LD) family and the binary relevance (BR) family. By emphasizing that these two categories refer to model families allows to view members from both families as adjustable methods which can be optimized and fine-tuned for MLTC tasks. However, importantly, when aiming for an optimal performance, it is commonly believed that methods incorporating label dependency information outperform those that transform the multi-label classification task into multiple binary classifications corresponding to BR methods.

As main result, we find that not only does a BR-CNN outperform other advanced BR methods, but it can also outperform the best methods from the LD family on targeted data. That means BR-CNN is not always the best but there are datasets for which this is the case. This finding challenges the prevailing view that methods from the LD family are universally superior to those from the BR family. Instead, it seems that neither the LD nor the BR family is universally superior but optimality is task dependent. While it remains to be seen how these results translate to other data types, our results serve as a cautionary message against dismissing BR methods. One reason we consider important for obtaining our results is to realize that all methods from the LD family, but also the BR family, are tunable and a comparison of both requires the selection of the best method from each family. Otherwise a comparison is unfair and featureless.

CRediT authorship contribution statement

Zhen Yang: Writing – review & editing, Writing – original draft, Visualization, Methodology. **Frank Emmert-Streib:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Conceptualization.

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

No data was used for the research described in the article.

Acknowledgments

Zhen Yang would like to thank the Faculty of Information Technology and Communication Sciences, Tampere University, Finland, for financial support.

References

[1] M.R. Boutell, J. Luo, X. Shen, C.M. Brown, Learning multi-label scene classification, Pattern Recognit. 37 (9) (2004) 1757–1771.
[2] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, H. Blockeel, Decision trees for hierarchical multi-label classification, Mach. Learn. 73 (2008) 185–214.
[3] G. Tsoumakas, I. Katakis, Multi-label classification: An overview, Int. J. Data Warehous. Min. (IJDWM) 3 (3) (2007) 1–13.
[4] F. Emmert-Streib, M. Dehmer, Taxonomy of machine learning paradigms: A data-centric perspective, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 12 (5) (2022) e1470.
[5] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni, J. Dong, Y. Zhao, S. Yan, HCP: A flexible CNN framework for multi-label image classification, IEEE Trans. Pattern Anal. Mach. Intell. 38 (9) (2015) 1901–1907.
[6] K. Trohidis, G. Tsoumakas, G. Kalliris, I.P. Vlahavas, et al., Multi-label classification of music into emotions, in: ISMIR, Vol. 8, 2008, pp. 325–330.
[7] T. Altaf, S.M. Anwar, N. Gul, M.N. Majeed, M. Majid, Multi-class Alzheimer's disease classification using image and clinical features, Biomed. Signal Process. Control 43 (2018) 64–74.
[8] Z. Yang, F. Emmert-Streib, Threshold-learned CNN for multi-label text classification of electronic health records, IEEE Access (2023) 1, http://dx.doi.org/10.1109/ACCESS.2023.3309157.
[9] Z. Barutcuoglu, R.E. Schapire, O.G. Troyanskaya, Hierarchical multi-label prediction of gene function, Bioinformatics 22 (7) (2006) 830–836.
[10] J. Du, Q. Chen, Y. Peng, Y. Xiang, C. Tao, Z. Lu, ML-Net: multi-label classification of biomedical texts with deep neural networks, J. Am. Med. Inform. Assoc. 26 (11) (2019) 1279–1285.
[11] D. Vilar, M.J. Castro, E. Sanchis, Multi-label text classification using multinomial models, in: Advances in Natural Language Processing: 4th International Conference, EsTAL 2004, Alicante, Spain, October 20-22, 2004. Proceedings 4, Springer, 2004, pp. 220–230.
[12] S. Spat, B. Cadonna, I. Rakovac, C. Gütl, H. Leitner, G. Stark, P. Beck, Multi-label text classification of German language medical documents, in: MedInfo, 2007, pp. 1460–1461.
[13] S. Godbole, S. Sarawagi, Discriminative methods for multi-labeled classification, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2004, pp. 22–30.
[14] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach. Learn. 85 (2011) 333–359.
[15] W. Cheng, E. Hüllermeier, K.J. Dembczynski, Bayes optimal multilabel classification via probabilistic classifier chains, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 279–286.
[16] J. Liu, W.-C. Chang, Y. Wu, Y. Yang, Deep learning for extreme multi-label text classification, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 115–124.
[17] S. Baker, A. Korhonen, Initializing neural networks for hierarchical multi-label text classification, in: BioNLP 2017, 2017, pp. 307–315.
[18] A. Pal, M. Selvakumar, M. Sankarasubbu, Multi-label text classification using attention-based graph neural network, 2020, arXiv preprint arXiv:2003.11644.
[19] C.-P. Tsai, H.-Y. Lee, Order-free learning alleviating exposure bias in multi-label classification, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 6038–6045.
[20] N. Liu, Q. Wang, J. Ren, Label-embedding bi-directional attentive model for multi-label text classification, Neural Process. Lett. 53 (2021) 375–389.
[21] X. Zhang, Q.-W. Zhang, Z. Yan, R. Liu, Y. Cao, Enhancing label correlation feedback in multi-label text classification via multi-task learning, 2021, arXiv preprint arXiv:2106.03103.
[22] N. Rastin, M.Z. Jahromi, M. Taheri, A generalized weighted distance k-nearest neighbor for multi-label problems, Pattern Recognit. 114 (2021) 107526.
[23] S. Kumar, N. Kumar, A. Dev, S. Naorem, Movie genre classification using binary relevance, label powerset, and machine learning classifiers, Multimedia Tools Appl. 82 (1) (2023) 945–968.
[24] P. Yang, X. Sun, W. Li, S. Ma, W. Wu, H. Wang, SGM: sequence generation model for multi-label classification, 2018, arXiv preprint arXiv:1806.04822.
[25] P.J. Hayes, S.P. Weinstein, CONSTRUE/TIS: A system for content-based indexing of a database of news stories, in: IAAI, Vol. 90, 1990, pp. 49–64.
[26] S. Gehrmann, F. Dernoncourt, Y. Li, E.T. Carlson, J.T. Wu, J. Welt, J. Foote Jr., E.T. Moseley, D.W. Grant, P.D. Tyler, et al., Comparing deep learning and concept extraction based methods for patient phenotyping from clinical narratives, PLoS One 13 (2) (2018) e0192360.
[27] D.D. Lewis, Y. Yang, T. Russell-Rose, F. Li, Rcv1: A new benchmark collection for text categorization research, J. Mach. Learn. Res. 5 (Apr) (2004) 361–397.

[28] G. Tsoumakas, I. Katakis, I. Vlahavas, Mining multi-label data, Data mining and knowledge discovery handbook (2010) 667–685.
[29] F. Emmert-Streib, Z. Yang, H. Feng, S. Tripathi, M. Dehmer, An introductory review of deep learning for prediction models with big data, Front. Artif. Intell. 3 (2020) 4.
[30] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. (TIST) 2 (3) (2011) 1–27.
[31] A. Elisseeff, J. Weston, A kernel method for multi-labelled classification, Adv. Neural Inf. Process. Syst. 14 (2001).
[32] M.-L. Zhang, ML-RBF: Rbf neural networks for multi-label learning, Neural Process. Lett. 29 (2009) 61–74.
[33] Z. Yang, M. Dehmer, O. Yli-Harja, F. Emmert-Streib, Combining deep learning with token selection for patient phenotyping from electronic health records, Sci. Rep. 10 (1) (2020) 1432.
[34] Z. Yang, F. Emmert-Streib, Threshold-learned CNN for multi-label text classification of electronic health records, IEEE Access (2023).
[35] H. Khataei Maragheh, F.S. Gharehchopogh, K. Majidzadeh, A.B. Sangar, A new hybrid based on long short-term memory network with spotted hyena optimization algorithm for multi-label text classification, Mathematics 10 (3) (2022) 488.
[36] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, J. Gao, Deep learning–based text classification: a comprehensive review, ACM Comput. Surv. (CSUR) 54 (3) (2021) 1–40.
[37] J. Nowak, A. Taspinar, R. Scherer, LSTM recurrent neural networks for short text and sentiment classification, in: Artificial Intelligence and Soft Computing: 16th International Conference, ICAISC 2017, Zakopane, Poland, June 11-15, 2017, Proceedings, Part II 16, Springer, 2017, pp. 553–562.
[38] A. Elnagar, R. Al-Debsi, O. Einea, Arabic text classification using deep learning models, Inf. Process. Manage. 57 (1) (2020) 102121.
[39] W. Yin, K. Kann, M. Yu, H. Schütze, Comparative study of CNN and RNN for natural language processing, 2017, arXiv preprint arXiv:1702.01923.
[40] B. Wang, C. Li, V. Pavlu, J. Aslam, A pipeline for optimizing f1-measure in multi-label text classification, in: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2018, pp. 913–918.
[41] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1746–1751, http://dx.doi.org/10.3115/v1/D14-1181, URL https://aclanthology.org/D14-1181.
[42] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. (ISSN: 0899-7667) 9 (8) (1997) 1735–1780, http://dx.doi.org/10.1162/neco.1997.9.8.1735, arXiv:https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf.
[43] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, 2014, arXiv preprint arXiv:1406.1078.
[44] A. Shewalkar, D. Nyavanandi, S.A. Ludwig, Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU, J. Artif. Intell. Soft Comput. Res. 9 (4) (2019) 235–245.
[45] Y. Fan, J. Liu, P. Liu, Y. Du, W. Lan, S. Wu, Manifold learning with structured subspace for multi-label feature selection, Pattern Recognit. 120 (2021) 108169.
[46] Y. Fan, B. Chen, W. Huang, J. Liu, W. Weng, W. Lan, Multi-label feature selection based on label correlations and feature redundancy, Knowl.-Based Syst. 241 (2022) 108256.
[47] Y. Huang, B. Giledereli, A. Köksal, A. Özgür, E. Ozkirimli, Balancing methods for multi-label text classification with long-tailed class distribution, 2021, arXiv preprint arXiv:2109.04712.
[48] P.-T. De Boer, D.P. Kroese, S. Mannor, R.Y. Rubinstein, A tutorial on the cross-entropy method, Ann. Oper. Res. 134 (2005) 19–67.
[49] Y. Yang, An evaluation of statistical approaches to text categorization, Inf. Retr. 1 (1–2) (1999) 69–90.
[50] Y. Yang, A study of thresholding strategies for text categorization, in: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2001, pp. 137–145.
[51] R.A. Reale, J.F. Brugge, J.C. Chan, Maps of auditory cortex in cats reared after unilateral cochlear ablation in the neonatal period, Dev. Brain Res. 34 (2) (1987) 281–290.
[52] P.O. Kanold, P. Kara, R.C. Reid, C.J. Shatz, Role of subplate neurons in functional maturation of visual cortical columns, Science 301 (5632) (2003) 521–525.
[53] A.F. Abate, L. Cimmino, J. Lorenzo-Navarro, An ablation study on part-based face analysis using a Multi-input convolutional neural network and semantic segmentation, Pattern Recognit. Lett. 173 (2023) 45–49.
[54] S. Sheikholeslami, M. Meister, T. Wang, A.H. Payberah, V. Vlassov, J. Dowling, Autoablation: Automated parallel ablation studies for deep learning, in: Proceedings of the 1st Workshop on Machine Learning and Systems, 2021, pp. 55–61.
[55] R. Wang, R. Ridley, W. Qu, X. Dai, et al., A novel reasoning mechanism for multi-label text classification, Inf. Process. Manage. 58 (2) (2021) 102441.
[56] K. Dembczyński, W. Waegeman, W. Cheng, E. Hüllermeier, On label dependence and loss minimization in multi-label classification, Mach. Learn. 88 (2012) 5–45.
[57] T.N. Rubin, A. Chambers, P. Smyth, M. Steyvers, Statistical topic models for multi-label document classification, Mach. Learn. 88 (2012) 157–208.
[58] S. Ji, L. Tang, S. Yu, J. Ye, A shared-subspace learning framework for multi-label classification, ACM Trans. Knowl. Discov. Data (TKDD) 4 (2) (2010) 1–29.
[59] S.-J. Huang, Z.-H. Zhou, Multi-label learning by exploiting label correlations locally, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 26, 2012, pp. 949–955.