Information Processing and Management 58 (2021) 102441

Contents lists available at ScienceDirect

Information Processing and Management

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

A novel reasoning mechanism for multi-label text classification

Ran Wang[a], Robert Ridley[a], Xi'ao Su[b], Weiguang Qu[b], Xinyu Dai[*,a]

[a] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
[b] Department of Computer Science and Technology, Nanjing Normal University, Nanjing 210023, China

ARTICLE INFO

Keywords:
Multi-label learning
Text classification
Label embedding
Iterative reasoning mechanism

2010 MSC:
00-01
99-00

ABSTRACT

The aim in multi-label text classification is to assign a set of labels to a given document. Previous classifier-chain and sequence-to-sequence models have been shown to have a powerful ability to capture label correlations. However, they rely heavily on the label order, while labels in multi-label data are essentially an unordered set. The performance of these approaches is therefore highly variable depending on the order in which the labels are arranged. To avoid being dependent on label order, we design a reasoning-based algorithm named Multi-Label Reasoner (ML-Reasoner) for multi-label classification. ML-Reasoner employs a binary classifier to predict all labels simultaneously and applies a novel iterative reasoning mechanism to effectively utilize the inter-label information, where each instance of reasoning takes the previously predicted likelihoods for *all labels* as additional input. This approach is able to utilize information between labels, while avoiding the issue of label-order sensitivity. Extensive experiments demonstrate that our method outperforms state-of-the art approaches on the challenging AAPD dataset. We also apply our reasoning module to a variety of strong neural-based base models and show that it is able to boost performance significantly in each case.

1. Introduction

Multi-label classification (MLC) is an important task in natural language processing and is applicable to many real-word scenarios, such as papers covering multiple disciplines, blog posts discussing multiple topics, comments revealing multiple emotions, etc. The aim of the MLC task is to assign multiple non-exclusive labels to each sample. For example, Table 1 shows an example of multi-label text classification. In this example, the text should be tagged with the labels: "basketball", "NBA" and "sport".

The naive approach to MLC is to transform it into multiple independent binary classification problems. This approach is known as Binary Relevance (BR) Gonçalves and Quaresma (2003), and it is widely used due to its simplicity. However, this method completely ignores the relevant information between labels. Intuitively, knowing some labels – such as "basketball" and "NBA" – should make it easier to predict the label "sport". Most researchers Liu, Chang, Wu, and Yang (2017); Read, Pfahringer, Holmes, and Frank (2011); Yang et al. (2018); Zhang and Zhou (2007) have also pointed out that it is beneficial, even necessary, to make use of the inter-label relationships for the task of MLC. There are several methods that attempt to accurately capture label correlations, among which Classifier Chains (CC), proposed in Read et al. (2011), may be the best-known method for MLC. The CC method links together multiple binary classifiers, with each classifier using the predictions from the previous classifiers as input. This method takes the possible label

* Corresponding author.
E-mail addresses: wangr@smail.nju.edu.cn (R. Wang), robertr@smail.nju.edu.cn (R. Ridley), sephiroth_sxa@163.com (X. Su), wggu_nj@163.com (W. Qu), daixinyu@nju.edu.cn (X. Dai).

https://doi.org/10.1016/j.ipm.2020.102441
Received 17 January 2020; Received in revised form 15 July 2020; Accepted 14 November 2020
Available online 29 November 2020

dependencies into account, but a major drawback of this approach is that different label orders can produce disparate results. Moreover, the chaining procedure means that CC cannot be parallelized, and thus a high computational cost is incurred when handling large datasets.

Nam, Loza Menc'ia, Kim, and Fürnkranz (2017); Yang et al. (2018) view label sets as label sequences and handle the MLC task through a neural network-based sequence-to-sequence (seq2seq) model. Compared with CC, which make binary decisions for each label sequentially, seq2seq models only explicitly predict positive labels. Therefore, the length of their decision chain is equal to the number of positive labels, not the number of all labels. This allows them to achieve better performance. However, these models depend heavily on how the labels are ordered. In multi-label data, the labels are essentially sets and are not necessarily ordered naturally. To cast label-sets into label-sequences, various methods have been explored: sorting heuristically Yang et al. (2018), by greedy algorithms, by dynamic programming Liu and Tsang (2015), or by reinforcement learning Yang, Luo, Ma, Lin, and Sun (2019). Previous experiment Liu and Tsang (2015); Qin, Li, Pavlu, and Aslam (2019); Yang et al. (2019, 2018) results have shown that the label order has a significant impact on learning and prediction. For example, if the label sequence in Table 1, were to begin with “sport”, it would be less useful for predicting other relevant labels.

To tackle the above issues, we propose Multi-Label Reasoner (ML-Reasoner), a reasoning-based algorithm, for the task of MLC. We employ a binary classifier for each label and *predict all labels simultaneously* to satisfy the unordered nature of the labels. CC and seq2seq models need to specify the order of the labels in advance by some criterion, such as alphabetical order. If we look at the instance in Table 1 for example, we can see that sorting the labels alphabetically would arrange them in the order of “basketball”, “football”, “sport”, “NBA”, and this is the sequence in which the CC and seq2seq algorithms would predict the labels. In contrast, ML-Reasoner approaches the problem by simultaneously calculating the probability of each label being positive; for instance, ML-Reasoner may give the probability of “NBA” being positive to be 0.9, “basketball” to be 0.7, “sport” to be 0.55 and “football” to be 0.3. Therefore, ML-Reasoner is able to avoid having to depend on label orders completely. To make effective use of the label correlations, we design a novel iterative reasoning mechanism, whereby the previously predicted likelihoods for *all* labels are used as additional features during reasoning. This method enables ML-Reasoner to refine the predictions during each round of reasoning. For example, consider the scenario in which high probabilities are assigned to the labels “NBA” and “basketball”; as a result, in the subsequent instance of reasoning, the label “sport” would also be assigned a higher probability.

In summary, our contributions are as follows:

1. We design a novel reasoning-based method ML-Reasoner, which performs multiple rounds of predictions, with each round of predictions using the previous round as additional input in order to refine and update them.
2. By using the previous set of predictions, the reasoning mechanism enables our model to utilize the relationships that exist between labels, while also avoiding the need to rely on label order.
3. Through thorough testing, we demonstrate that ML-Reasoner is able to outperform the current state-of-the-art MLC models.
4. We apply our reasoning mechanism to a number of strong neural-based base models and show that in each case, performance is significantly improved. For instance, on the challenging AAPD dataset, applying our Reasoner component to CNN and BERT models yields an absolute value improvement of 2% in micro-F_1 score.

The remainder of this paper is organized as follows: In section 2, we provide an overview of the relevant literature and give context to our work. In Section 3, we introduce ML-Reasoner, a novel and effective model for multi-label text classification. This model not only utilizes the relationships between labels, but also avoids being dependent on the order of the labels. Section 4 contains results for experiments carried out on the RCV1-V2 and AAPD datasets, which demonstrate our model's effectiveness in MLC. The detailed ablation study in Section 5 highlights the irreplaceable role of the reasoning mechanism applied by our approach. We also include further analyses to verify that ML-Reasoner avoids being dependent on the order of the labels. In addition, our analyses demonstrate how our model utilizes the relationships between labels to refine its predictions. Finally, we conclude in Section 6.

2. Background

In this section, we discuss well-known methods for multi-label text classification, mainly including problem transformation methods and algorithm adaption methods. In addition, since neural networks are closely related to our method, we also separately discuss the application of neural networks to multi-label classification problems.

2.1. Problem Transformation Approaches

Problem transformation-based multi-label classification algorithms aim to convert the original problem into a set of binary classification problems or multi-class classification problems, which are then handled with traditional classification algorithms. These binary classification-based transformation methods can be grouped into two categories, depending on whether these binary classification tasks are obtained by a one-versus-one strategy or a one-versus-all strategy.

Binary Relevance (BR, Gonçalves & Quaresma (2003)) is one of the most well-known transformation-based methods. BR employs a one-versus-all strategy to decompose the multi-label classification problem into multiple independent binary classification problems equivalent to the number of labels in the dataset. A binary classifier is then trained for each label. The classifier outputs are combined to produce the predicted labelset. While it is a straightforward and simple method, BR completely dismisses the potential relationships between labels by training each binary classifier independently. In order to model label correlations, 2BR Tsoumakas et al. (2009)

stacks BR in to two levels, with the first level learning the BR model, and the second level using the outputs from the first level to learn a meta-model. BR+ Cherman, Metz, and Monard (2012) is another approach to incorporate label dependencies. In BR+, each binary classifier uses the non-target labels as additional inputs for training. The drawbacks of BR are also mitigated by the Classifier Chain (CC) method proposed in Read et al. (2011). The CC trains multiple binary classifiers as sub-tasks. In contrast to how the BR algorithm operates, the CC links the binary classifiers in a randomly-ordered chain. For a given example, each binary classifier incorporates the labels predicted by the previous classifiers as additional information. Since the order in which the classifiers in a CC are linked influences the information that is given to each classifier, the performance of a CC is variable depending on the label order. To remedy this drawback, an ensemble of classifier chains (ECC) is proposed in Read et al. (2011), which employs a set of CCs where binary classifiers are sorted in a different order in each CC. However, the computational efficiency of this method Dembczynski, Cheng, and Hüllermeier (2010); Read et al. (2011) is an issue for datasets with a large number of labels or samples.

RPC (Ranking by Pairwise Comparison, Hüllermeier, Fürnkranz, Cheng, & Brinker (2008)) is based on a one-versus-one strategy. It converts the multi-label dataset with k single labels to $k(k-1)/2$ binary datasets, considering all possible label pairs. To improve this method, a Calibrated Label Ranking algorithm (CLR, Fürnkranz, Hüllermeier, Loza Menc'ia, & Brinker (2008)) introduces a fictional label to automatically adjust the cut threshold. Compared to the BR method, these algorithms increase the classification performance, albeit at the cost of a longer training time.

The Label Powerset (LP) method proposed in Boutell, Luo, Shen, and Brown (2004) is one other transformation approach. The authors treat each possible label combination as a unique class identifier and apply multi-class classifiers to approach the task as an intermediate multi-class classification problem. While this approach indirectly considers label correlations, the number of different label combinations in datasets with a high number of labels can be incredibly large, with some combined classes having few training examples. Another obstacle is that an LP classifier cannot predict label combinations that don't appear in the training set. To avoid the previous problems, a pruning method named Pruned Sets (PS) is proposed in Read (2008), which prunes the infrequent atomic labels. Meanwhile, the Ensemble of Pruned Sets (EPS, Read, Pfahringer, & Holmes (2008)) is based on the PS method and is able to predict label combinations that have not been seen in the training data. Though this procedure improves the generalization ability of the classifier, the dataset loses some of its multi-label structure. In addition, there are some proposals that combine the BR and LP transformations Tenenboim-Chekina, Rokach, and Shapira (2010) to perform multi-label classification.

2.2. *Algorithm Adaption Approaches*

As an alternative approach, algorithm adaption-based methods extend traditional classification techniques to tackle multiple labels directly.

Multilabel C4.5 (ML-C4.5, De Comité, Gilleron, & Tommasi (2003)) can be seen as a generalization of the well-known C4.5 algorithm Ruggieri (2002). In order to directly deal with multiple labels, the leaves of the tree in ML-C4.5 are samples associated with a set of labels, and ML-C4.5 employs a multi-label entropy measure to take into account the non-member probability of an instance for a label. There are also some other proposals based on trees to directly tackle multi-label classification problem, such as ADTBoost.MH Schapire and Singer (1999), ML-Tree Wu, Ye, Zhang, Chow, and Ho (2015), LaCova Al-Otaibi, Kull, and Flach (2014), etc.

Several multi-label algorithms have been adapted from Support Vector Machines (SVM) Vapnik (2000). Elisseeff and Weston (2001) proposed a multi-label algorithm named Rank-SVM. Rank-SVM employs a new metric, which is a lineal approximation of hamming loss, as a training object to train SVMs. To improve the efficiency and performance of Rank-SVM, Rank-CVM Xu (2013) uses a core vector machine instead of an SVM, whose analytical solution is immediate and much more efficient than that of SVMs. Meanwhile, SCRank-SVM Wang, Feng, Sun, Chen, and Chen (2014) reformulates the calculus of the decision boundary of SVMs to reduce the computational complexity.

There are also several multi-label methods adapted from instance-based learning algorithms Aha, Kibler, and Albert (1991), which do not build an explicit model through a training process. Zhang and Zhou (2007) present the best-known instance-based MLC algorithm named "Multi-Label kNN" (MLkNN), which internally works as a BR classifier. MLkNN assigns a set of labels to a given instance by computing a priori and conditional probabilities for each label. Based on the MLkNN algorithm, two similar MLC methods are proposed in Cheng and Hüllermeier (2009), named IBLR-ML and IBLR-ML+. They use bayesian techniques to consider the label within the k nearest neighbors of the new instance as extra attributes and logistic regression is employed to capture the inter-dependencies between the labels.

Algorithm adaption-based approaches rely on traditional algorithms, which is a largely studied problem in data mining. However, most of them perform poorly or are computationally intractable when dealing with high-order label correlations.

2.3. *Neural Network-Based Approaches*

Neural networks have been used to great effect in MLC, with current state-of-the-art results coming from neural-based methods. The first multi-label-adapted neural-network-based model proposed in the literature is BP-MLL Zhang and Zhou (2006b). BP-MLL employs a fully-connected neural network and introduces a pairwise ranking loss function to take into account that each sample contains several labels. I-BP-MLL Grodzicki, Mandziuk, and Wang (2008) proposes producing a custom threshold for each label, instead of a global threshold. Nam, Kim, Loza Menc'ia, Gurevych, and Fürnkranz (2014) reports that it is better to utilize cross-entropy loss instead of ranking loss to train neural networks. Kurata, Xiang, and Zhou (2016) utilize convolutional neural networks to capture each pattern of label co-occurrence. The CNN-RNN proposed by Chen, Ye, Xing, Chen, and Cambria (2017) employs a convolutional and recurrent-neural-network ensemble to represent both the local and global semantics and to model high-order label correlations. Liu

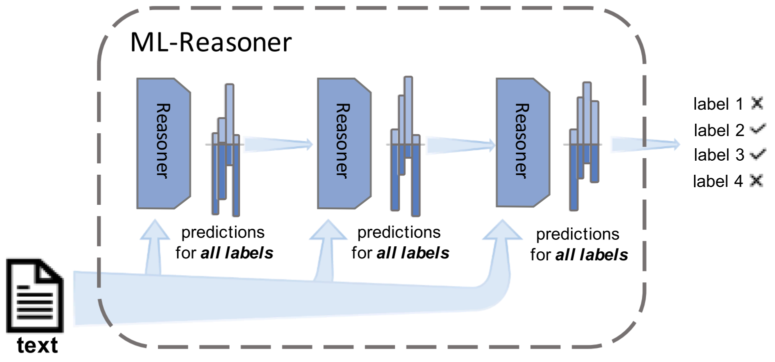
ML-Reasoner
Reasoner
Reasoner
Reasoner
predictions for *all labels*
predictions for *all labels*
predictions for *all labels*
label 1 ✗
label 2 ✓
label 3 ✓
label 4 ✗
text

Fig. 1. Illustration of Multi-Label Reasoner. The arrows indicate inputs or outputs. The internal *Reasoner* not only takes account of the text but also the label predictions from the previous reasoning instance. During each iteration, the Reasoner predicts *all* labels simultaneously, which enables ML-Reasoner to avoid being dependent on label order. The Reasoner in this figure iterates three times. See Section 3 for further details.

et al. (2017) present a new deep learning approach named XML-CNN for extreme multi-label text classification, which uses a dynamic max pooling scheme and a hidden bottleneck layer for better representations of documents. Peng et al. (2018) convert each document into a graph of words, and then they leverage a graph convolution operation to capture long-distance semantics. Nam et al. (2017) utilize powerful recurrent neural networks, especially encoder-decoder frameworks, to capture the label dependencies. To improve performance, Yang et al. (2018) propose SGM, which equips a seq2seq framework with an attention mechanism. Most recently, Yang et al. (2019) present a deep reinforced sequence-to-set model named seq2set to model the unordered nature of labelsets. Qin et al. (2019) also provide a new objective to effectively obtain probabilities of label sets instead of label sequences. Xiao, Huang, Chen, and Jing (2019) propose a label-specific attention network to learn the document representation, which takes advantage of label semantic information to determine the semantic connection between the labels and a document to construct a label-specific document representation. Meanwhile, the Bidirectional Encoder Representation from Transformers (BERT) introduced by Devlin, Chang, Lee, and Toutanova (2019) have created state-of-the-art models for a wide range of tasks.

3. Multi-Label Reasoner

In this section, we introduce Multi-Label Reasoner (ML-Reasoner) in detail. We first define some necessary notations and formulate the multi-label text classification task. Then, the framework of ML-Reasoner is introduced in Section 3.2, and Section 3.3 presents our novel reasoning module.

3.1. Notations

Let $D = \{(x_i, y_i)\}_{i=1}^{N}$ denote the dataset, which contains N documents with corresponding labels $y = \{0, 1\}^k$ with k being the total number of label classes and y representing either the presence or absence of a label. Let $x_i = \{w_1, \cdots, w_p, \cdots, w_n\}$ indicate the i-th document containing n words with w_p being the p-th word in the document and n being the number of words in document. The multi-label text classification task requires training a classifier f to assign the most relevant labels to a piece of text.

3.2. Framework

An overview of our proposed framework is shown in the Figure 1. As discussed above, labels in multi-label datasets are given as sets, not necessarily ordered in a natural way. To avoid needing to sort labels, we employ a reasoner to capture the semantics of the document and make a prediction for each label. Meanwhile, since information between labels is necessary for MLC, we take the label predictions from the previous round of reasoning as additional input to the reasoner to refine the predictions. This method passes label information between different iterations, allowing *ML-Reasoner* to take the label correlations into account and thus overcoming the label independence problem. It is worth noting that there are some key differences between this reasoning-based method and the chaining method applied by the CC or the seq2seq approaches. Each classifier in CC uses the prediction for previous labels as additional input, and classifiers in the seq2seq-based models use the predictions for previous *positive* labels as additional input. In contrast, *ML-Reasoner* takes into account the possible inter-label relationships by using the predictions for *all labels* of the previous iteration as additional input. This reasoning process is repeated a number of times and the labels predicted during the final iteration are assigned to the document.

The reasoning process is described in Algorithm 1. Formally, at time t, the *Reasoner* g takes the word sequence x and the label prediction $z_{t-1} = (z_{t-1,1}, \cdots, z_{t-1,k})$ of previous time $t-1$ as the input and output label relevant probabilities z_t (Line 5 in Algorithm 1). Because this work aims to explore whether the reasoning mechanism can effectively utilize the relationships between labels, the reasoner g does not use any prior knowledge of the labels, i.e. z_0=**none** (Line 3 in Algorithm 1). It is easy to see that the design of the *Reasoner* is the core of this algorithm, and the reasoner iteration number T (line 4 in Algorithm 1) is an important hyper-parameter.

```
Input: Dataset $D$, Reasoner iteration number $T$
repeat
    foreach instance $(x, y) \in D$ do
        $z$ = none;
        foreach $t$ in $1 \ldots T$ do
            $z = g(x, z)$;
        Update the model according to $loss(z, y)$ ;
until reaching stop criterion;
return a multi-label classifier $f$;
```

Algorithm 1. Multi-Label Reasoner

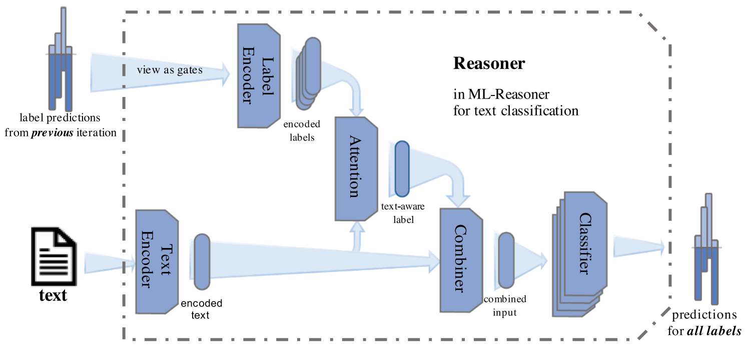

Fig. 2. The Reasoner module in ML-Reasoner for multi-label text classification. The arrows indicate inputs or outputs. The Reasoner consist of five components: Text Encoder, Label Encoder, Attention, Combiner, and Classifiers. Note the Reasoner uses the label predictions from the previous round of reasoning to utilize label correlations.

3.3. Details of Reasoner

Figure 2 shows the detailed inner-architecture of the *Reasoner* module. This reasoner not only reads in texts to mine semantic information, but also uses the label predictions from the previous round of reasoning to utilize label correlations.

Given the word sequence $x = (w_1, \cdots, w_n)$, the ground-truth $y = \{0, 1\}^k$, and label predictions from the previous round of reasoning $t - 1$, i.e., $z_{t-1} = (z_{t-1,1}, \cdots, z_{t-1,k})$ with $z_{t-1,j} \in [0, 1]$, each component in the reasoner functions as follows.

Text Encoder reads word sequences and produces a representation for each document, which is responsible for extracting text features. It consists of Text Embedding and CNN Encoder.

Text Embedding transforms words in documents into dense vectors. It first converts the characters in each word into character vectors and then applies a CNN Kim (2014) with max-pooling to obtain character-derived embeddings. Each word is also mapped to a pre-trained word vector. The two embeddings are then concatenated and passed on to the CNN Encoder:

$$\vec{w}_i = \text{TextEmbedding}(w_i) \in \mathbb{R}^{D_1} \tag{1}$$

where $i \in \{1, \cdots, n\}$.

CNN Encoder Kim (2014) with max-pooling is employed as an encoder to obtain sentence representations:

$$\vec{x} = \text{CNN-with-maxpool}\left(\left[\vec{w}_1, \cdots, \vec{w}_n\right]\right) \in \mathbb{R}^{D_2} \tag{2}$$

where [,] represents the vector stack operation. Then a linear layer is applied:

$$\vec{x}_{\text{encoded}} = W_1^T \vec{x} + b_1 \in \mathbb{R}^{D_3} \tag{3}$$

where $W_1 \in \mathbb{R}^{D_2 \times D_3}$ and $b_1 \in \mathbb{R}^{D_3}$ are parameters to be learned.

Label Encoder converts the previous reasoning result for each label $z_{t-1,j} \in \mathbb{R}$ into a corresponding label representation $\vec{l}_{j\text{encoded}} \in \mathbb{R}^{D_4}$. Unlike a one-hot representation, the low-dimensional dense vector representation can reflect the correlation between the labels. Therefore, we employ a *Label Embedding* to map each label into a randomly initialized vector with length D_4:

$$\vec{l}_j = \text{LabelEmbedding}(l_j) \in \mathbb{R}^{D_4} \tag{4}$$

where $j \in \{1, \cdots, k\}$. If a high probability is assigned to a particular label of a document, the label should also affect the current prediction result to a greater extent. Intuitively, we view the reasoning results $z_{t-1,j} \in \mathbb{R}$ as a gate and utilize the gate-mechanism applied by GRUs Cho et al. (2014) to combine the label representations $\vec{l}_j$:

$$\vec{l}_{j\text{encoded}} = z_{t-1,j} \vec{l}_j \in \mathbb{R}^{D_4} \tag{5}$$

Since the text is often only related to some labels, we use an attention mechanism to further mine the relationship between them. We compute the weight α_j to the j-th label at first:

$$\begin{aligned} s_j &= W_2\left[\vec{x}_{\text{encoded}}; \vec{l}_{j\text{encoded}}\right] + b_2 \\ \alpha_j &= \frac{\exp(s_j)}{\sum_{i=1}^{k} \exp(s_i)} \end{aligned} \tag{6}$$

Table 1
An example of multi-label classification for text. In the example, labels relevant to the text are listed under the heading *positive labels* and the one irrelevant label under *negative labels*.

Text	This article is about a game between Houston Rockets and Los Angeles Lakers.
positive labels	basketball, NBA, sport
negative labels	football

Table 2
Summary of the datasets. *Total Samples* and *Label Sets* represent the number of samples and size of the label set respectively. *Words/Sample* is the average number of words per sample. *Card* and *Dens* are the indices introduced in Herrera et al. (2016). *Dens* stands for *label density*, a normalized value that is calculated by dividing the label cardinality by the total number of labels in the dataset.

Dataset	Total Samples	Label Sets	Words/Sample	Dens
AAPD	55,840	54	163.43	0.045
RCV1-V2	804,414	103	123.94	0.031

where $W_2 \in \mathbb{R}^{D_3+D_4}$ and $b_2 \in \mathbb{R}$ are internal parameters of the model and $[;]$ is vector concatenation. Afterwards, the label-aware text vector $\overrightarrow{l}_{\text{attention}}$ is obtained:

$$\overrightarrow{l}_{\text{attention}} = \sum_{j=1}^{k} \alpha_j \overrightarrow{l}_{j_{\text{encoded}}} \in \mathbb{R}^{D_4} \tag{7}$$

Inspired by residual networks He, Zhang, Ren, and Sun (2016), we fuse together the information obtained from both the label and text features:

$$\overrightarrow{x}_{\text{combined}} = \left[\overrightarrow{x}_{\text{encoded}}; \overrightarrow{l}_{\text{attention}}\right] \in \mathbb{R}^{D_3+D_4} \tag{8}$$

Classifiers with *the same structure but different parameters* are applied to make predictions for each label. The output of the j-th classifier of this iteration t is defined as follows:

$$z_{t,j} = \text{sigmoid}\left(W_{4j}^{T}\tanh\left(W_{3j}^{T}\overrightarrow{x}_{\text{combined}} + b_{3j}\right) + b_{4j}\right) \in \mathbb{R} \tag{9}$$

where $W_{3j} \in \mathbb{R}^{(D_3+D_4)\times D_5}$, $b_{3j} \in \mathbb{R}^{D_5}$, $W_{4j} \in \mathbb{R}^{D_5}$ and $b_4 \in \mathbb{R}$ are all internal parameters. $z_{t,j}$ indicates the probability that the j-th label is positive.

Loss Function We use *Binary Cross Entropy Loss* (BCELoss) to calculate the loss for the prediction $\overrightarrow{z}_t$ and the ground-truth $\overrightarrow{y}$:

$$\ell\left(\overrightarrow{z}_t, \overrightarrow{y}\right) = \frac{1}{k}\sum_{i=1}^{k} -\left[y_i \log z_{t,i} + (1-y_i)\log\left(1-z_{t,i}\right)\right] \tag{10}$$

Correspondingly, with the reasoner iteration number set to T, the average loss for the dataset D is defined as follows:

$$loss = \frac{1}{TN}\sum_{i=1}^{T}\sum_{j=1}^{N}\ell\left(\overrightarrow{z}_{ij}, \overrightarrow{y}_j\right) \tag{11}$$

4. Experiment Settings

In this section, we introduce details regarding the two datasets, baselines, metrics and detailed settings that were used in our experiments.

4.1. Datasets

We conduct experiments on two datasets:

Arxiv Academic Paper Dataset (AAPD) is created by Yang et al. (2018). The task for this dataset is to judge which of a set of academic disciplines each paper belongs to according to the abstracts.
Reuters Corpus Volume I (RCV1-V2) is provided by Lewis, Yang, Rose, and Li (2004). The task for this dataset is to match the desensitized news stories to the relevant topics.

Table 2 summarizes some key statistics regarding the AAPD and RCV1-V2 datasets. From the table, we can see that compared to the

RCV1-V2 datase, documents in the AAPD dataset contain more words and have a higher label density on average, but the size of this dataset is smaller Herrera, Charte, Rivera, and del Jesús (2016). These statistics reveal that processing the AAPD dataset is more challenging than the RCV1-V2 dataset because the model needs to be more powerful to mine more accurate information from less data. We follow Yang et al. (2018)'s way of dividing training, validation and test sets. [1]

4.2. Evaluation Metrics

As with previous researchers Yang et al. (2019); Yang et al. (2018); Zhang and Zhou (2006a), we adopt hamming loss, micro-precision, micro-recall and micro-F_1 as our primary evaluation metrics. We also use precision, recall, F_1 and macro-F_1 Herrera et al. (2016) to aid in our analyses.

Hamming-Loss Schapire and Singer (1998) calculates scores according to instance-label pair prediction errors made by the model, with errors being termed as instances of predicting irrelevant labels or failing to predict relevant labels.
Micro-F_1 Manning, Raghavan, and Schütze (2008) is calculated after counting the number of true positives, true negatives, false negatives, and false positives. Higher frequency labels usually contribute more to the final measure.

4.3. Baselines

Our model is compared with the following baselines:

BR Gonçalves and Quaresma (2003) transforms the MLC task into multiple binary classification tasks.
LP Boutell et al. (2004) views the MLC task as a multi-class classification problem.
CC Read et al. (2011) applies a sequence of binary classification tasks to solve the MLC task.
CNN-RNN Chen et al. (2017) uses CNNs and RNNs to obtain local and global semantics, and also model the relationships between labels.
SGM Yang et al. (2018) treats the MLC task as a sequence-generation task and uses seq2seq as a multi-class classifier.
LSAN Xiao et al. (2019) utilizes label semantic information and a self-attention mechanism to determine the semantic connection between labels and documents.
seq2set Yang et al. (2019) trains a sequence-to-sequence model via reinforcement learning, where reward feedback is designed to be independent of the label order.

ML-Reasoner can be applied on top of any text encoder. Therefore, we also test and compare performance for the following models with and without the addition of our Reasoner mechanism:

CNN applies a convolutional neural network to learn a dense feature matrix before performing classification.
LSTM applies a long short-term memory network to consider the sequential structure of the text as well as alleviate the issue of exploding and vanishing gradients.
BERT uses Bidirectional Encoder Representations from Transformers trained on a large-scale corpus. BERT has achieved state-of-the-art performance on a wide spectrum of natural language tasks.

After each of the above neural-based models obtains a text representation, the labels are predicted with Equation 9, and BCELoss is applied as the loss function in each case.

4.4. Model Settings

For BR, LP and CC, we use the average of each word's representation vector as the text feature, and the embedding matrix is initialized with the 300 dimension GloVe[2] Pennington, Socher, and Manning (2014) vectors pre-trained on the Common Crawl corpus. The vectors are made trainable during the training stage. A logistic regression model is adopted as a base classifier for BR, LP and CC. For CNN-RNN, SGM, LSAN and seq2set, we re-implement all models with the same hyper-parameters as specified in the corresponding original papers.

The ML-Reasoner model and other neural text encoders are implemented with AllenNLP Gardner et al. (2017), an open-source library built on PyTorch[3] for NLP research. We employ BERT-base-uncased model as our BERT baseline. To match the dimensions of BERT, the input dimension of the multi-layer perceptron is also 768. We use BERTAdam as the optimizer for BERT, setting the learning rate to 5-e5 and the warmup coefficient to 0.1. CNN, LSTM and ML-Reasoner use the same pre-trained GloVe vectors that are used in BR, CC and LP, and the vectors are set to be trainable as well. Meanwhile, the character embeddings are 300-dimension vectors which are initialized randomly. Likewise, the label embeddings of ML-Reasoner are also set to 300 dimensions and initialized

[1] These data are available online at https://github.com/lancopku/SGM.
[2] http://nlp.stanford.edu/data/glove.840B.300d.zip
[3] https://github.com/pytorch/pytorch

Table 3
Results on both datasets for the baseline and ML-Reasoner models. *HL, P, R* and *F1* refer to hamming loss, micro-precision, micro-recall and micro-F$_1$ respectively. The "-" symbol indicates metrics for which lower scores are more desirable, whereas the "+" symbol indicates that higher scores are more preferable. The hyperparameter *T* is set to 2 for each ML-Reasoner. The best performance is highlighted in bold.

Models	HL(-)	P(+)	R(+)	F1(+)
BR	0.0266	0.710	0.632	0.669
CC	0.0256	0.758	0.629	0.688
LP	0.0255	0.745	0.655	0.697
CNN-RNN	0.0261	0.726	0.669	0.697
SGM	0.0251	0.748	0.675	0.710
LSAN	0.0239	0.771	0.666	0.715
seq2set	0.0255	0.735	0.679	0.706
CNN	0.0259	0.728	0.676	0.700
LSTM	0.0254	**0.776**	0.611	0.683
BERT	0.0258	0.722	0.690	0.705
ML-Reasoner	**0.0238**	0.761	0.684	0.720
ML-Reasoner$_{LSTM}$	0.0255	0.746	0.655	0.698
ML-Reasoner$_{BERT}$	0.0248	0.726	**0.718**	**0.722**

Performance of models on AAPD test set.

Models	**HL(-)**	**P(+)**	**R(+)**	**F1(+)**
BR	0.0086	0.904	0.816	0.858
CC	0.0087	0.887	0.828	0.857
LP	0.0087	0.896	0.824	0.858
CNN-RNN	0.0087	0.878	0.838	0.858
SGM	0.0079	0.885	**0.860**	0.872
LSAN	0.0078	0.899	0.845	0.871
seq2set	**0.0076**	0.900	0.858	**0.878**
CNN	0.0083	0.881	0.851	0.866
LSTM	0.0081	0.888	0.850	0.869
BERT	0.0081	0.890	0.846	0.868
ML-Reasoner	0.0081	**0.912**	0.847	**0.878**
ML-Reasoner$_{LSTM}$	0.0079	0.897	0.845	0.870
ML-Reasoner$_{BERT}$	0.0079	0.890	0.852	0.871

Performance of models on RCV1-V2 test set.

randomly. Adamax Kingma and Ba (2014) is our optimizer, with the learning rate set to 2e-3. To alleviate over-fitting, we apply variational dropout Srivastava, Hinton, Krizhevsky, Sutskever, and Salakhutdinov (2014) and clip the gradients Pascanu, Mikolov, and Bengio (2013) to the maximum norm of 5.0. For more details please refer to Table 7 in the appendix.

5. Results and Analysis

5.1. *ML-Reasoner v.s. Baselines Analysis*

The experiment results of the ML-Reasoner and baseline models on the AAPD and RCV1-V2 test sets are shown in tables 3a and 3 b respectively. From table 3a, we can see that ML-Reasoner outperforms all baselines by a significant margin on the more challenging AAPD dataset. In particular, the hamming loss score of 0.0238 is a 5.2% improvement on the SGM score of 0.0251, the best among the baselines; and the micro-F$_1$ score shows a 1.0% (absolute) improvement over SGM. ML-Reasoner also proves effective when applied to the various text encoders that were tested. Specifically, on the AAPD dataset, CNN, LSTM and BERT receive a 2% absolute value increase in micro-F$_1$ score when our Reasoner mechanism is applied.

Table 3 b shows that as the size of the dataset increases (RCV1 contains roughly 4.5 times the number of instances as AAPD), the performance difference between different models decreases. However, it is still evident that the Reasoner mechanism brings about an improvement in performance over all base models, and the ML-Reasoner's micro-F$_1$ score of 0.878 is the highest among all baselines.

5.2. *Sensitivity to Label Order*

To verify that our approach is insensitive to the label order, we randomly shuffle the order of label sequences. Since the LP, LSAN, CNN, LSTM and BERT models are similar to BR, i.e., they predict all labels *simultaneously*, rather than *sequentially*, we only include the model BR as a baseline in this experiment. Meanwhile, we choose the CNN as ML-Reasoner's base classifier due to its speed and performance. Table 4 demonstrates the performance of various models on the label-shuffled AAPD dataset. The experiment results reveal that:

Table 4
Results on the label-shuffled AAPD. "↓" indicates that the model's performance has degraded. The best performance is highlighted in bold.

Models	HL (-)	P (+)	R (+)	F1 (+)
BR	0.0266 (↓ 0.0%)	0.710 (↓ 0.0%)	0.632 (↓ 0.0%)	0.669 (↓ 0.0%)
CC	0.0268 (↓ 4.70%)	0.739 (↓ 2.46%)	0.620 (↓ 1.44%)	0.674 (↓ 1.91%)
CNN-RNN	0.0267 (↓ 2.30%)	0.726 (↓ 0.04%)	0.649 (↓ 3.08%)	0.685 (↓ 1.65%)
SGM	0.0254 (↓ 1.20%)	0.742 (↓ 2.09%)	0.663 (↓ 5.76%)	0.700 (↓ 2.06%)
seq2set	0.0256 (↓ 0.77%)	0.730 (↓ 0.78%)	0.668 (↓ 1.56%)	0.698 (↓ 1.19%)
ML-Reasoner	**0.0238** (↓ 0.0%)	**0.761** (↓ 0.0%)	**0.684** (↓ 0.0%)	**0.720** (↓ 0.0%)

Table 5
Some examples from the AAPD dataset showing the difference of performance as a result of reasoning. *True Labelset* is the set of actual labels for the example; *before Reasoning* is the prediction result before undergoing reasoning; *after Reasoning* is the prediction result after applying reasoning. For the texts of examples please refer to Table 8 in the appendix. The performance differences are represented by ↑, which indicates an increase in performance, and ↓, which indicates a decrease in performance. For brevity, we do not display the text data, which can be found in the appendix. * "cond-mat" stands for "cont-mat.stat-mech". "physics" stands for "physics.soc-ph".

ID	True Labelset	before Reasoning	after Reasoning	Δ
1	cs.NI, cs.SY, math.OC	cs.NI, cs.SY	cs.NI, cs.SY, *math.OC*	↑
2	cs.DB, cs.PL	cs.DB, *cs.LO*, cs.PL	cs.DB, cs.PL	↓
3	cs.SI, physics*, cond-mat*	cs.SI, physics, *cs.CC*	cs.SI, physics, *cond-mat*	↑
4	cs.SC, cs.LO, cs.SY	*cs.LO*, cs.MS,cs.SC	cs.MS, cs.SC	↓
5	cs.SI, physics	cs.SI, *physics*, cs.IR	cs.SI, cs.IR	↓

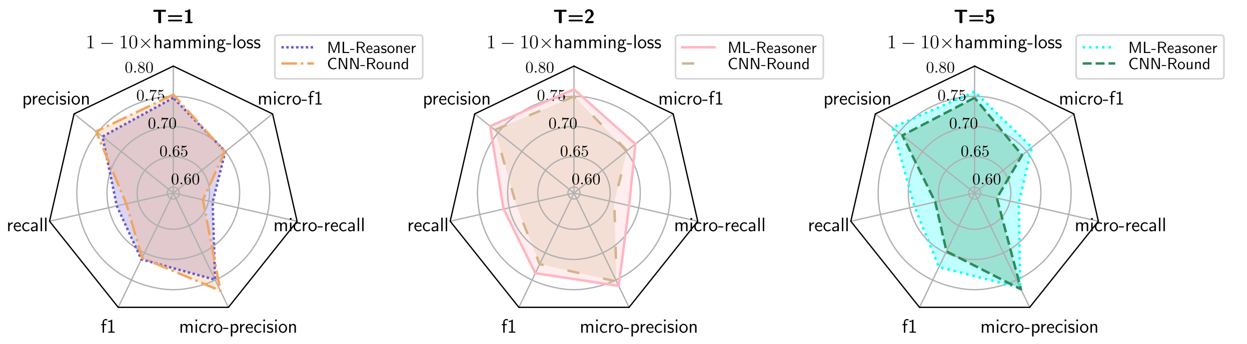

Fig. 3. Performance radar maps on the AAPD test set for ML-Reasoner ($T = 1,2,5$) and CNN-Round ($T = 1,2,5$) respectively. Note that for the sake of appearance and the radar chart principle of *the larger the area, the better the performance*, the metric displayed on the graph is $1 - 10 \times$ hamming-loss instead of hamming-loss.

1. The performance of CC and other sequence-to-sequence models, i.e. CNN-RNN and SGM, declines drastically. This decline is due to the fact that these models rely heavily on the label order while training and predicting. However, the labels are not necessarily in their natural order.
2. BR and our proposed ML-Reasoner are not affected. This is because they predict all labels *simultaneously*, rather than *sequentially*. Meanwhile, in contrast to BR-like models, our model takes into account the possible label dependencies by utilizing predictions from the previous time step as extra features, which enables ML-Reasoner to outperform all baselines.
3. Compared with the model seq2set, which uses reinforcement learning, our ML-Reasoner is able to avoid the label-order dependency issue in a more complete fashion.

5.3. Ablation Study

To understand the importance of the Reasoner module in our approach, we compare ML-Reasoner with an ablated version of our methods named *CNN-Round*. Specifically, CNN-Round uses the same text encoder and classifiers as ML-Reasoner. It is also carried out T

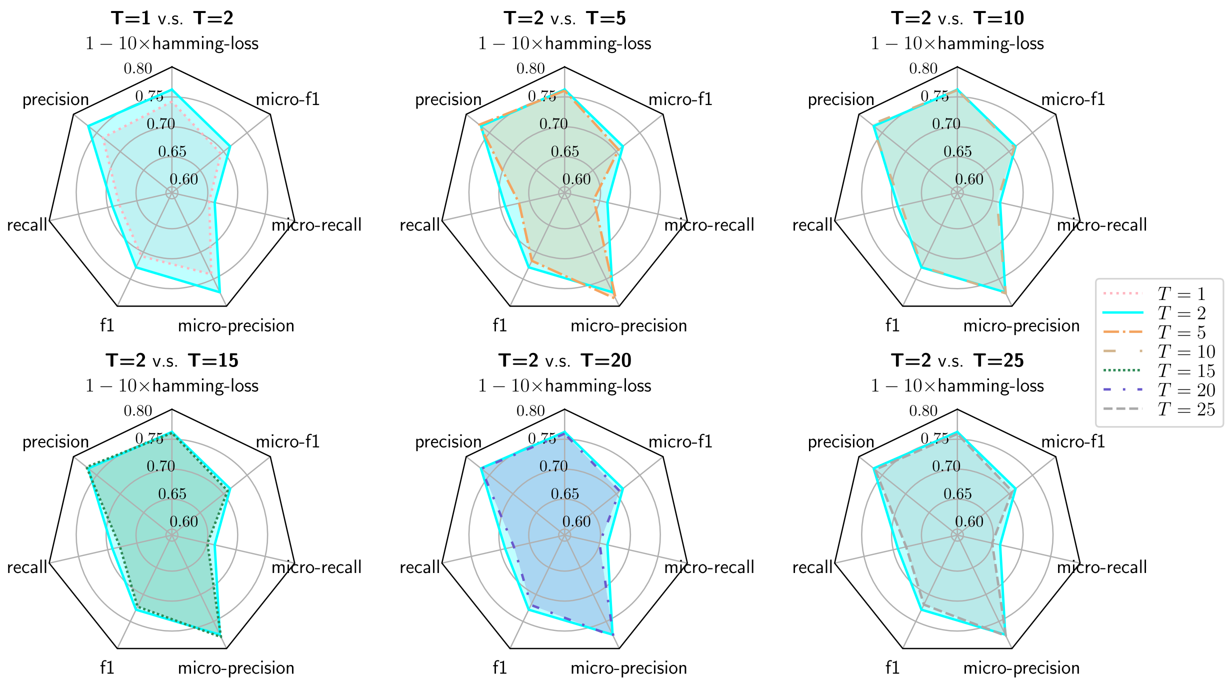

Fig. 4. Comparison of performance on AAPD test set for ML-Reasoner with differing values for the number of rounds of reasoning. As before, 1 −10 × hamming-loss is used instead of hamming-loss.

times for every round of predictions and optimized by the average loss in Eq. 11. The hyper-parameters of both the CNN-Round model and the ML-Reasoner are entirely same. However, CNN-Round differs from ML-Reasoner in that it does not employ the *label encoder* and the *attention mechanism*, which means it cannot utilize predictions for all labels in order to refine them. It is worth noting that CNN-Round is also different from the CNN baseline, where CNN-Round carries out T times for every round of predictions and optimized by the average loss in Eq. 11 while CNN just makes one time prediction and is optimized by the BCELoss.

The comparison between ML-Reasoner and CNN-Round for different values of T value is shown in Fig. 3. As can be seen from the figure, when $T = 1$, the scores for all metrics for both ML-Reasoner and CNN-Round are very similar. This is because ML-Reasoner has yet to perform any reasoning. However, when $T = 2, 5$, ML-Reasoner outperforms CNN-Round by a large margin for most metrics. The result reveals that the reasoning process in the ML-Reasoner is critical to the task of multi-label classification.

5.4. The Impact of Reasoner Iteration Number

Another question is the relation between the performance of ML-Reasoners and reasoner iteration number T.

For this purpose, we conducted experiments on the AAPD dataset with the ML-Reasoner T value set to $1, 2, 5, 10, 15, 20$ and 25. We conducted five experiments for each value and then recorded the average performance. The performance comparisons for each different T value are shown in Fig. 4.

Fig. 4 shows that the area representing the performance for $T = 1$ is entirely covered by the area representing $T = 2$, while the areas for $T = 5, 10, 15, 20$ and 25 essentially coincide with each other. Considering that a larger area covered in the radar chart indicates a better performance, this result illustrates that the ML-Reasoner with $T = 2$ outperforms one with $T = 1$ and further rounds of reasoning do not bring an improvement in performance.

5.5. The Role of Reasoner

Explainability in learning system has been receiving more and more attention Makni, Abdelaziz, and Hendler (2020). Therefore, to further understand the role that the Reasoner module plays in multi-label text classification, we have selected some examples from the AAPD dataset for which the predictions have changed as a result of reasoning, as shown in Table 5.

In the first example, before reasoning takes place, the model predicts the label “cs.NL” and “cs.SY” to be relevant. Once reasoning has taken place, however, the model adds the label “Math.OC” to achieve a more accurate set of predictions. In the second example, the

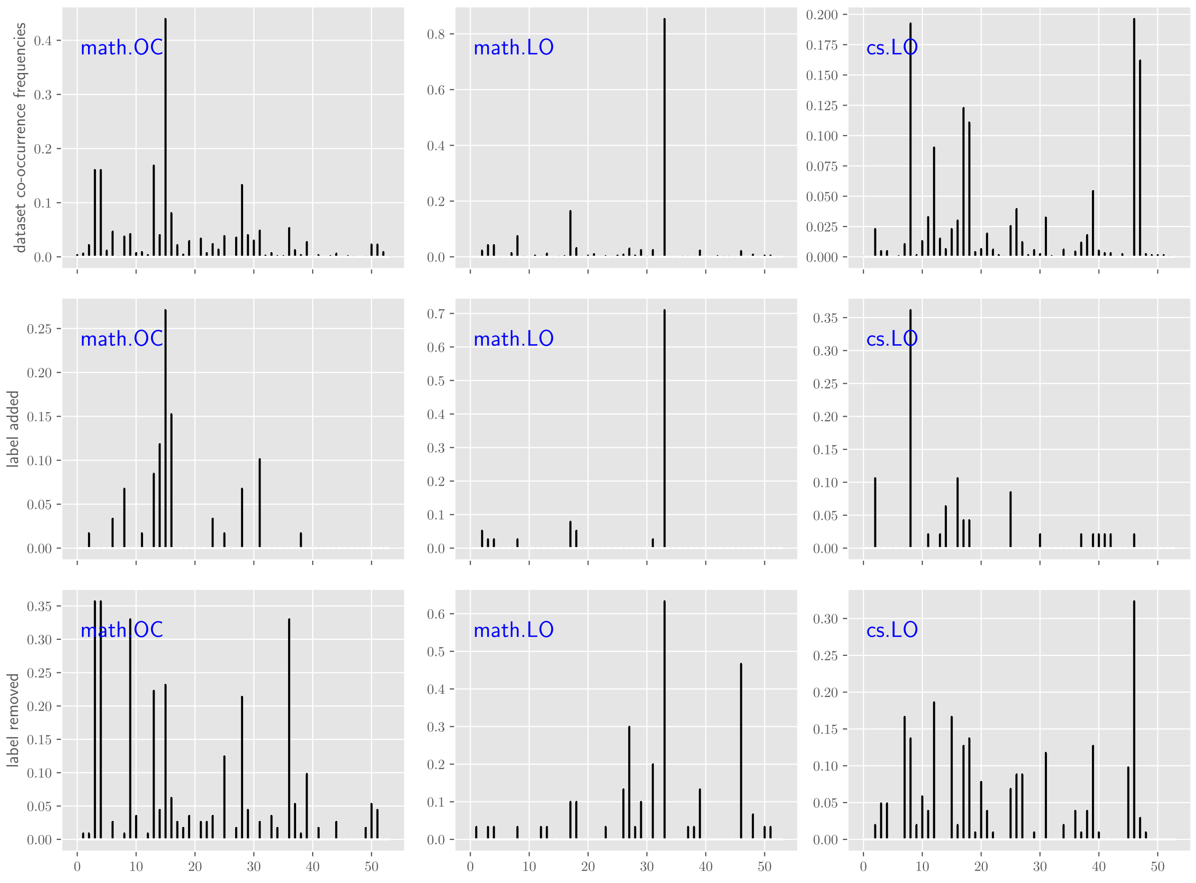

Fig. 5. Co-occurrence frequencies for labels in the AAPD dataset. The first row of graphs shows the co-occurrence frequencies (displayed on the y axis) between the labels “math.OC”, “math.LO”, “cs.LO” and all other labels in the dataset (x axis). The second row displays the pre-reasoning co-occurrence frequencies when these three labels were added to the prediction set through reasoning, and the third row displays the pre-reasoning co-occurrence frequencies when they were removed from the prediction set through reasoning.

Table 6
Summary of metric differences between the final predictions and the intermediate results without any reasoning. Δ ↑ represents the number of label classes whose performance *improved*; Δ ↓ represents the number of label classes whose performance *declined*. *Mean* refers to the average value in terms of difference in performance.

Metric	Δ ↑	Δ ↓	Mean
F_1	40	14	+ 0.0038
precision	45	9	+ 0.0137
recall	39	15	− 0.0073

model removes an irrelevant label from its set of predictions following the reasoning process. In the third example, the model both removes an irrelevant label from and adds a relevant label to its prediction set following reasoning. In some cases the Reasoner also leads to inferior predictions, such as with examples 4 and 5 in Table 5.

In order to verify whether the changes shown in the examples above are due to consideration for the relationships between labels, we include a graph displaying the frequency of label co-occurrence in the AAPD training set, as shown in Fig. 5. In the top row of graphs the co-occurrence frequencies between the labels “math.OC”, “math.LO”, “cs.LO” and other labels in the datastet are displayed. The second and third rows of graphs display the co-occurrence frequencies when these labels have been added and removed by the reasoning mechanism. As can be seen, the true co-occurrence frequencies (first-row graphs) for labels in the dataset and the pre-reasoning co-occurrence frequencies for labels that were subsequently added by the reasoning mechanism (second-row graphs) display a high degree of similarity. In contrast, the difference between the true co-occurrence frequencies and the pre-reasoning co-occurrence frequencies for labels that were subsequently removed by the reasoning mechanism (third-row graphs) display a much lower degree of similarity. This is possibly due to the reasoning mechanism being able to add labels when the joint distribution between two labels resembles the distribution in the dataset and being able to remove labels when the distribution is further from that in the dataset.

In order to quantitatively explore whether reasoning affects the behavior of the model in the ways described above, we collect the prediction results of the model on the training set and calculate the pre-reasoning and post-reasoning performance differences across various metrics, such as precision, recall and F_1.

The summary of these differences is shown in Table 6. Here, we can see that for almost all label classes, precision increased and for some label classes recall slightly decreased. These changes are consistent with the example given in Table 5.

To summarize the above result, the Reasoner can effectively utilize the relevant information between labels: adding relevant labels and removing irrelevant labels.

6. Conclusion and Future work

In this paper, we present ML-Reasoner, a novel reasoning mechanism for multi-label text classification. The proposed model makes a prediction for each label simultaneously to avoid the need to depend on label order, as with sequence-to-sequence models. Meanwhile, it can utilize high-order correlations among labels through a novel reasoning mechanism. Results from extensive experiments show that ML-Reasoner is effective in capturing inter-label dependencies and outperforms competitive baselines on a challenging multi-label dataset. Additional analyses further verify that the iterative reasoning mechanism in ML-Reasoner does respect the unordered nature of labels and it also plays an important role in MLC. We also study some empirical cases to understand the role that the Reasoner plays in multi-label text classification. Since our method does not explicitly introduce the type of relationships that exist between labels, such as hierarchies, further consideration could go in to utilizing this external knowledge more effectively, which we also plan to do in future work.

CRediT authorship contribution statement

Ran Wang: Conceptualization, Methodology, Software, Investigation, Writing - original draft, Writing - review & editing, Visualization. **Robert Ridley:** Writing - review & editing. **Xi'ao Su:** Software. **Weiguang Qu:** Supervision. **Xinyu Dai:** Supervision, Project administration, Funding acquisition.

Acknowledgements

This work is supported by National Key R&D Program of China (2018YFB1005102) and Natural Science Foundation of China (No. 61976114 and No.61936012).

Appendix A

Table 7
Details setting of ML-Reasoner. The dropout for the output of the text embedding and label encoder are set to a rate of 0.2, and the dropout for the first layer output of the MLP is set to a rate of 0.5.

Module	Hyper-parameter	Value
Text Embedding	filter sizes	3, 4, 5
	number of each filter	all 100
	dimension of output vectors, D_1	600
CNN Encoder	filter sizes	2, 3, 4, 5
	number of each filter	all 100
	dimension of a sentence-level vector, D_2	400
	dimension of output vectors, D_3	300
Label Encoder	dimension of label embeddings, D_4	300
Classifiers	dimension of intermediate vectors, D_5	200
Trainer	number of epochs	300
	patience for early stopping	5
Adamax Optimizer	learning rate	2e−3
	coefficients used for computing running averages of gradient and its square, βs	0.9, 0.999
	weight decay (L2 penalty)	0

Table 8
Texts of examples shown in Table 5.

ID	Text
1	we consider a generalized processing system having several queues, where the available service rate combinations are fluctuating over time due to reliability and availability variations the objective is to allocate the available resources, and corresponding service rates, in response to both workload and service capacity considerations, in order to maintain the long term stability of the system the service configurations are completely arbitrary, including negative service rates which represent forwarding and service induced cross traffic we employ a trace based trajectory asymptotic technique, which requires minimal assumptions about the arrival dynamics of the system we prove that cone schedules, which leverage the geometry of the queueing dynamics, maximize the system throughput for a broad class of processing systems, even under adversarial arrival processes we study the impact of fluctuating service availability, where resources are available only some of the time, and the schedule must dynamically respond to the changing available service rates, establishing both the capacity of such systems and the class of schedules which will stabilize the system at full capacity the rich geometry of the system dynamics leads to important insights for stability, performance and scalability, and substantially generalizes previous findings the processing system studied here models a broad variety of computer, communication and service networks, including varying channel conditions and cross traffic in wireless networking, and call centers with fluctuating capacity the findings have implications for bandwidth and processor allocation in communication networks and workforce scheduling in congested call centers
2	provenance is information recording the source, derivation, or history of some information provenance tracking has been studied in a variety of settings however, although many design points have been explored, the mathematical or semantic foundations of data provenance have received comparatively little attention in this paper, we argue that dependency analysis techniques familiar from program analysis and program slicing provide a formal foundation for forms of provenance that are intended to show how (part of) the output of a query depends on (parts of) its input we introduce a semantic characterization of such dependency provenance, show that this form of provenance is not computable, and provide dynamic and static approximation techniques
3	divorced individuals face complex situations when they have children with different ex partners, or even more, when their new partners have children of their own in such cases, and when kids spend every other weekend with each parent, a practical problem emerges is it possible to have such a custody arrangement that every couple has either all of the kids together or no kids at all ? we show that in general, it is not possible, but that the number of couples that do can be maximized the problem turns out to be equivalent to finding the ground state of a spin glass system, which is known to be equivalent to what is called a weighted max cut problem in graph theory, and hence it is np complete
4	one of the most frequently used models for understanding human navigation on the web is the markov chain model, where web pages are represented as states and hyperlinks as probabilities of navigating from one page to another predominantly, human navigation on the web has been thought to satisfy the memoryless markov property stating that the next page a user visits only depends on her current page and not on previously visited ones this idea has found its way in numerous applications such as google 's pagerank algorithm and others recently, new studies suggested that human navigation may better be modeled using higher order markov chain models, i e, the next page depends on a longer history of past clicks yet, this finding is preliminary and does not account for the higher complexity of higher order markov chain models which is why the memoryless model is still widely used in this work we thoroughly present a diverse array of advanced inference methods for determining the appropriate markov chain order we highlight strengths and weaknesses of each method and apply them for investigating memory and structure of human navigation on the web our experiments reveal that the complexity of higher order models grows faster than their utility, and thus we confirm that the memoryless model represents a quite practical model for human navigation on a page level however, when we expand our analysis to a topical level, where we abstract away from specific page transitions to transitions between topics, we find that the memoryless assumption is violated and specific regularities can be observed we report results from experiments with two types of navigational datasets (goal oriented vs free form) and observe interesting structural differences that make a strong argument for more contextual studies of human navigation in future work
5	we present abstraction techniques that transform a given non linear dynamical system into a linear system or an algebraic system described by polynomials of bounded degree, such that, invariant properties of the resulting abstraction can be used to infer invariants for the original system the abstraction techniques rely on a change of basis transformation that associates each state variable of the abstract system with a function involving the state variables of the original system we present conditions under which a given change of basis transformation for a non linear system can define an abstraction furthermore, the techniques developed here apply to continuous systems defined by ordinary differential equations (odes), discrete systems defined by transition systems and hybrid systems that combine continuous as well as discrete subsystems the techniques presented here allow us to discover, given a non linear system, if a change of bases transformation involving degree bounded polynomials yielding an algebraic abstraction exists if so, our technique yields the resulting abstract system, as well this approach is further extended to search for a change of bases transformation that abstracts a given non linear system into a system of linear differential inclusions our techniques enable the use of analysis techniques for linear systems to infer invariants for non linear systems we present preliminary evidence of the practical feasibility of our ideas using a prototype implementation

References

Aha, D. W., Kibler, D. F., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine learning, 6*, 37–66.

Al-Otaibi, R., Kull, M., & Flach, P. A. (2014). Lacova: A tree-based multi-label classifier using label covariance as splitting criterion. *13th international conference on machine learning and applications, ICMLA 2014* (pp. 74–79).

Boutell, M. R., Luo, J., Shen, X., & Brown, C. M. (2004). Learning multi-label scene classification. *Pattern Recognition, 37*(9), 1757–1771.

Chen, G., Ye, D., Xing, Z., Chen, J., & Cambria, E. (2017). Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. *2017 international joint conference on neural networks, IJCNN 2017* (pp. 2377–2383).

Cheng, W., & Hüllermeier, E. (2009). Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning, 76*(2-3), 211–225.

Cherman, E. A., Metz, J., & Monard, M. C. (2012). Incorporating label dependency into the binary relevance framework for multi-label classification. *Expert Systems with Applications, 39*(2), 1647–1655.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP 2014* (pp. 1724–1734).

De Comité, F., Gilleron, R., & Tommasi, M. (2003). Learning multi-label alternating decision trees from texts and data. *International workshop on machine learning and data mining in pattern recognition* (pp. 35–49). Springer.

Dembczynski, K., Cheng, W., & Hüllermeier, E. (2010). Bayes optimal multilabel classification via probabilistic classifier chains. *Proceedings of the 27th international conference on machine learning, 2010* (pp. 279–286).

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies, NAACL-HLT 2019* (pp. 4171–4186).

Elisseeff, A., & Weston, J. (2001). A kernel method for multi-labelled classification. *Advances in neural information processing systems 14 [neural information processing systems: Natural and synthetic, NIPS 2001* (pp. 681–687).

Fürnkranz, J., Hüllermeier, E., Loza Menc'ia, E., & Brinker, K. (2008). Multilabel classification via calibrated label ranking. *Machine Learning, 73*(2), 133–153.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N. F., … Zettlemoyer, L. S. (2017). *Allennlp: A deep semantic natural language processing platform. abs/1803.07640*.

Gonçalves, T., & Quaresma, P. (2003). A preliminary approach to the multilabel classification problem of portuguese juridical documents. *Portuguese conference on artificial intelligence* (pp. 435–444).

Grodzicki, R., Mandziuk, J., & Wang, L. (2008). Improved multilabel classification with neural networks. *Parallel problem solving from nature - PPSN x, 10th international conference dortmund, 2008, proceedings* (pp. 409–416).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE conference on computer vision and pattern recognition, CVPR 2016* (pp. 770–778).

Herrera, F., Charte, F., Rivera, A. J., & del Jesús, M. J. (2016). *Multilabel Classification - Problem Analysis, Metrics and Techniques*. Springer.

Hüllermeier, E., Fürnkranz, J., Cheng, W., & Brinker, K. (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence, 172*(16-17), 1897–1916.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP 2014* (pp. 1746–1751).

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR, abs/1412.6980*.

Kurata, G., Xiang, B., & Zhou, B. (2016). Improved neural network-based multi-label classification with better initialization leveraging label co-occurrence. *NAACL HLT 2016, the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 521–526).

Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research, 5*, 361–397.

Liu, J., Chang, W., Wu, Y., & Yang, Y. (2017). Deep learning for extreme multi-label text classification. *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval, 2017* (pp. 115–124).

Liu, W., & Tsang, I. W. (2015). On the optimality of classifier chain for multi-label classification. *Advances in neural information processing systems 28: Annual conference on neural information processing systems 2015* (pp. 712–720).

Makni, B., Abdelaziz, I., & Hendler, J. A. (2020). Explainable deep RDFS reasoner. *CoRR, abs/2002.03514*.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Nam, J., Kim, J., Loza Menc'ia, E., Gurevych, I., & Fürnkranz, J. (2014). Large-scale multi-label text classification - revisiting neural networks. *Machine learning and knowledge discovery in databases - european conference, ECML PKDD 2014* (pp. 437–452).

Nam, J., Loza Menc'ia, E., Kim, H. J., & Fürnkranz, J. (2017). Maximizing subset accuracy with recurrent neural networks in multi-label classification. *Advances in neural information processing systems 30: Annual conference on neural information processing systems 2017* (pp. 5413–5423).

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *Proceedings of the 30th international conference on machine learning, ICML 2013* (pp. 1310–1318).

Peng, H., Li, J., He, Y., Liu, Y., Bao, M., Wang, L., … Yang, Q. (2018). Large-scale hierarchical text classification with recursively regularized deep graph-cnn. *Proceedings of the 2018 world wide web conference on world wide web, WWW 2018* (pp. 1063–1072).

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP 2014* (pp. 1532–1543).

Qin, K., Li, C., Pavlu, V., & Aslam, J. A. (2019). Adapting RNN sequence prediction model to multi-label set prediction. *Proceedings of the 2019 conference of the north american chapter of the association for computational linguistics: Human language technologies, NAACL-HLT 2019* (pp. 3181–3190).

Read, J. (2008). A pruned problem transformation method for multi-label classification*, 143150. Proceedings 2008 new zealand computer science research student conference*.

Read, J., Pfahringer, B., & Holmes, G. (2008). Multi-label classification using ensembles of pruned sets. *Proceedings of the 8th IEEE international conference on data mining (icdm), 2008* (pp. 995–1000).

Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. *Machine Learning, 85*(3), 333–359.

Ruggieri, S. (2002). Efficient C4.5. *IEEE transactions on knowledge and data engineering, 14*(2), 438–444.

Schapire, R. E., & Singer, Y. (1998). Improved boosting algorithms using confidence-rated predictions. *Proceedings of the eleventh annual conference on computational learning theory, COLT 1998* (pp. 80–91).

Schapire, R. E., & Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning, 37*(3), 297–336.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*(1), 1929–1958.

Tenenboim-Chekina, L., Rokach, L., & Shapira, B. (2010). Identification of label dependencies for multi-label classification. *Working notes of the second international workshop on learning from multi-label data* (pp. 53–60).

Tsoumakas, G., Dimou, A., Spyromitros, E., Mezaris, V., Kompatsiaris, I., & Vlahavas, I. (2009). Correlation-based pruning of stacked binary relevance models for multi-label learning. *Proceedings of the 1st international workshop on learning from multi-label data* (pp. 101–116).

Vapnik, V. (2000). The Nature of Statistical Learning Theory. In *Statistics for Engineering and Information Science*. Springer.

Wang, J., Feng, J., Sun, X., Chen, S., & Chen, B. (2014). Simplified constraints rank-svm for multi-label classification. *Pattern recognition - 6th chinese conference, CCPR 2014* (pp. 229–236).

Wu, Q., Ye, Y., Zhang, H., Chow, T. W. S., & Ho, S. (2015). ML-TREE: A tree-structure-based approach to multilabel learning. *IEEE transactions on neural networks and learning systems, 26*(3), 430–443.

Xiao, L., Huang, X., Chen, B., & Jing, L. (2019). Label-specific document representation for multi-label text classification. *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing, EMNLP-IJCNLP 2019* (pp. 466–475).

Xu, J. (2013). Fast multi-label core vector machine. *Pattern Recognition, 46*(3), 885–898.

Yang, P., Luo, F., Ma, S., Lin, J., & Sun, X. (2019). A deep reinforced sequence-to-set model for multi-label classification. *ACL 2019* (pp. 5252–5258).

Yang, P., Sun, X., Li, W., Ma, S., Wu, W., & Wang, H. (2018). SGM: sequence generation model for multi-label classification. *Proceedings of the 27th international conference on computational linguistics* (pp. 3915–3926).

Zhang, M., & Zhou, Z. (2006a). Multi-label neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering, 18*(10), 1338–1351.

Zhang, M., & Zhou, Z. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern recognition, 40*(7), 2038–2048.

Zhang, M.-L., & Zhou, Z.-H. (2006b). Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering, 18*(10), 1338–1351.