healthcare

MDPI

Technical Note

A Curriculum Batching Strategy for Automatic ICD Coding with Deep Multi-Label Classification Models

Yaqiang Wang [1,2,*], Xu Han [1,2], Xuechao Hao [3,4,*], Tao Zhu [3,4] and Hongping Shu [1,2]

[1] College of Software Engineering, Chengdu University of Information Technology, Chengdu 610225, China
[2] Sichuan Key Laboratory of Software Automatic Generation and Intelligent Service, Chengdu University of Information Technology, Chengdu 610225, China
[3] Department of Anesthesiology, West China Hospital, Sichuan University, Chengdu 621005, China
[4] The Research Units of West China, Chinese Academy of Medical Sciences, West China Hospital, Sichuan University, Chengdu 621005, China
* Correspondence: yaqwang@cuit.edu.cn (Y.W.); aneshxc@163.com (X.H.)

Abstract: The International Classification of Diseases (ICD) has an important role in building applications for clinical medicine. Extremely large ICD coding label sets and imbalanced label distribution bring the problem of inconsistency between the local batch data distribution and the global training data distribution into the minibatch gradient descent (MBGD)-based training procedure for deep multi-label classification models for automatic ICD coding. The problem further leads to an overfitting issue. In order to improve the performance and generalization ability of the deep learning automatic ICD coding model, we proposed a simple and effective curriculum batching strategy in this paper for improving the MBGD-based training procedure. This strategy generates three batch sets offline through applying three predefined sampling algorithms. These batch sets satisfy a uniform data distribution, a shuffling data distribution and the original training data distribution, respectively, and the learning tasks corresponding to these batch sets range from simple to complex. Experiments show that, after replacing the original shuffling algorithm-based batching strategy with the proposed curriculum batching strategy, the performance of the three investigated deep multi-label classification models for automatic ICD coding all have dramatic improvements. At the same time, the models avoid the overfitting issue and all show better ability to learn the long-tailed label information. The performance is also better than a SOTA label set reconstruction model.

Keywords: automatic ICD coding; curriculum learning; minibatch gradient descent; deep learning

Citation: Wang, Y.; Han, X.; Hao, X.; Zhu, T.; Shu, H. A Curriculum Batching Strategy for Automatic ICD Coding with Deep Multi-Label Classification Models. *Healthcare* **2022**, *10*, 2397. https://doi.org/10.3390/healthcare10122397

Academic Editor: Joaquim Carreras

Received: 17 September 2022
Accepted: 25 November 2022
Published: 29 November 2022

1. Introduction

The International Classification of Diseases (ICD) is a healthcare classification system initiated by the World Health Organization [1]. It is mainly used for uniformly coding diseases, symptoms, processes, injuries and other features contained in electronic medical records (EMRs) [2]. The coding results are widely used for epidemiological studies [3], the billing and reimbursement for medical services [4] and diagnostic information retrieval [5].

The ICD coding task requires professionals to master both medical knowledge and the ICD coding system. The coding process is time-consuming and laborious. In addition, different professionals may have different understandings of the complex ICD coding system. This may also lead to inconsistent coding results. Consequently, automatic ICD coding based on machine learning techniques has become an interesting research area in recent years [6,7].

Automatic ICD coding is usually regarded as a multi-label classification problem. The performance of automatic ICD coding is directly affected by extremely large label sets and imbalanced label distribution [8,9]. Recently, researchers have challenged these issues, mainly from two perspectives. One is reconstructing the label sets (ReLS) through

designing different label-combining strategies [7,8,10]. The new label sets consist of multi-granularity labels. Another is building multi-level label hierarchical classifiers based on hierarchical joint learning mechanisms [9,11]. The main idea of these methods is to reduce the label space dimension and alleviate the imbalanced label distribution issue.

Deep multi-label classification models are widely used for automatic ICD coding, and the minibatch gradient descent (MBGD) is a widely used optimization algorithm for training the models [7]. In each iteration during a training procedure, the gradient updates of the model parameters are approximated, based on the examples in one batch. The studies show that a larger batch size results in a lower frequency of gradient updates and better gradient update approximation, but it may hurt generalization and result in finding poorer local optima [12].

In this paper, we further find that, during training, deep multi-label classification models for automatic ICD coding, which split the training data into batches with a simple but commonly used shuffling algorithm in MBGD, always result in the local data distribution of the batches (or the local batch data distribution) being inconsistent with the global training data distribution (see Figure 1). This phenomenon causes the local statistics of batches to vary substantially from the global statistics of the training data [13]. It will also hurt the model performance and generalizability (see Table 1).

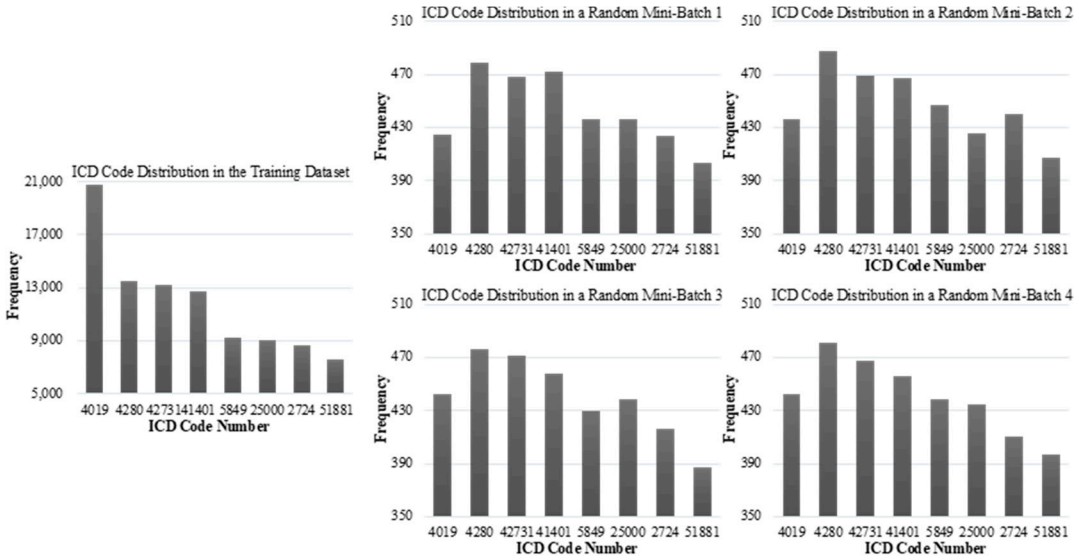

Figure 1. Comparing the global training data distribution (**left**) with sampled local batch data distributions (**right**). The right four charts are drawn based on four batches randomly chosen from one model training procedure. The model trained was based on the original MBGD, and the batch size is 500.

Table 1. Three representative deep multi-label classification models, TextCNN, TextRNN and TextRCNN for automatic ICD coding; all show signs of the overfitting issue. The algorithms obtained decent performance on the training data (top) but observed a large performance gap between training data and test data.

Models	TextCNN	TextRNN	TextRCNN
$F1_{micro}$ on Training Data	0.695	0.789	0.624
$F1_{micro}$ on Test Data	0.325	0.282	0.317

To challenge the inconsistency problem, we propose a simple and effective curriculum batching strategy to conveniently replace the shuffling algorithm for splitting the training data into batches. Assuming that the uniform data distribution (UDD) is easier to be learned by deep multi-label classification models for automatic ICD coding than the shuffling data

distribution (SDD), and the SDD is easier to be learned than the original imbalanced training data distribution (IDD), the proposed strategy applies three specific sampling algorithms to generate three batch sets with the three types of mentioned data distributions. The models then iterate each batch set, in turn, and learn the ICD coding experience from the easiest batch set to the hardest batch set.

A series of experimental results show that the proposed strategy can effectively avoid the overfitting issue and dramatically improve the automatic ICD coding performance compared with the three representative deep multi-label classification models and one state-of-the-art ReLS method for automatic ICD coding. Moreover, the improved models also have a better ability to learn the long-tailed label information. To the best of our knowledge, this is the first work to introduce a curriculum learning method for automatic ICD coding with deep multi-label classification models. In summary, the contribution of this paper is fourfold:

- This paper finds that the shuffling algorithm in the MBGD always causes the local data distribution of batches to be inconsistent with the global training data distribution, which hurts the model's performance and generalizability;
- This paper alleviates the problems of poor generalization ability and low classification accuracy of the deep learning model in the field of ICD automatic coding;
- This paper greatly improves the learning ability of the automatic ICD coding model for data with long-tailed labels;
- This paper introduces the curriculum learning method into automatic ICD coding of deep multi-label classification model for the first time.

2. Related Work

2.1. Automatic ICD Coding

Automatic ICD coding is a widely discussed, hot research topic in the field of medicine and healthcare [6,7]. Many statistical machine learning methods have been applied to challenge this problem [6,14]. In recent years, with the widespread success of deep learning, it has also been introduced to solve the problem of automatic ICD coding [7,15]. The automatic ICD coding problem is formulated as a multi-label classification problem [16,17].

Extremely large label sets and imbalanced data distribution are two main issues directly impacting coding performance. Mullenbach presented a Convolutional Attention for Multilabel Classification (CAML) model incorporating an attention technique with convolutional neural networks [18]. Sadoughi extended this approach by adding a maximum pooling layer across all the multiple convolution layers [19]. To further improve the performance of CAML, Li integrating a multi-filter CNN architecture with a residual CNN architecture, which is called a multi-filter residual convolutional neural network [16]. Bhutto present a Deep Recurrent Convolutional Neural Network with Hybrid Pooling (DRCNN-HP), which considers the different lengths as well as the dependency of the ICD code-related text chunks [20]. Another current solution idea is to reduce the label space dimensions to alleviate the imbalanced data distribution. The label set reconstruction method [7,8,10] and the multi-label hierarchical classifier-based method [9,11] are two representative approaches. Differently from the previous studies, in this paper we explored further, from the perspective of how these problems affect the local model training procedure.

2.2. Mini-Batch Gradient Descent

Mini-batch gradient descent is the widely used optimization algorithm to train deep neural models [21,22], including the deep multi-label classification models for automatic ICD coding [7]. How to effectively use MBGD to train models is always a concern in machine learning and other related fields [23,24]. There are many aspects of research, including the effect of batch normalization [25], batch size and learning rate settings [12,26], etc. We studied the inconsistency problem between the local batch data distribution and the global training data distribution and attempted to avoid the risk of overfitting and to improve the model generalization ability.

2.3. Curriculum Learning

Elman first proposed a "starting small strategy" approach to learn a model with a curriculum [27]. Two methods were developed, including an incremental input method and an incremental memory method. The motivation behind these methods was that compound sentences make it a little difficult for neural networks to learn grammatical concepts in the early phases of training. Therefore, these methods make the models learn gradually, by varying the percentage of simple and compound sentences used during training or constraining the model's capacity in the early phases of training [27].

Bengio et al. [28] revisited Elman's approach and formalized the approach of starting training on easy examples first and then gradually introducing more complex examples during the training. This type of approach was named as "curriculum learning", and it can achieve significant improvements in generalization. The core of curriculum learning is to develop a curriculum strategy to train the models from simple to complex.

Consequently, we proposed a simple and effective curriculum batching strategy for automatic ICD coding with deep multi-label classification models in order to solve the inconsistency problem mentioned earlier, improve the model generalization ability and avoid the risk of overfitting. The curriculum batching strategy generates and sequentially utilizes three types of batch sets composed of examples from easy-to-learn to hard-to-learn.

3. Methodology

In this section, we first review the original MBGD-based training procedure for deep multi-label classification models for automatic ICD coding. Then, we introduce the proposed curriculum batching strategy and how to integrate it conveniently into the MBGD-based training procedure, followed by the description of three specific sampling algorithms for obtaining batch sets satisfying the UDD, SDD and IDD.

3.1. MBGD-Based Training Procedure

The MBGD-based training procedure for deep multi-label classification models for automatic ICD coding is shown in Figure 2. This procedure consists of two modules: the data batch process and model training. In module one, the training data set $\mathcal{D}$ is divided into several mini-batches $(b_1, b_2, \ldots, b_{K-1}, b_K)$, which constitute the batch set $\mathcal{B}$. Second, each mini-batch b is sent to module two in turn. Every example $(\boldsymbol{x}, \boldsymbol{y})$ in mini-batch b contains an electronic case $\boldsymbol{x}$ and its corresponding ICD codes $\boldsymbol{y}$, a real example $(\boldsymbol{x}, \boldsymbol{y})$ is shown in Figure 3. Third, the examples are sent to the model training module. The deep multi-label ICD coding model uses the electronic case $\boldsymbol{x}$ to predict its ICD codes and compares the predicted output with the real output y. Finally, module 2 calculates the losses for each sample and updates the model. The procedure is repeated until the model converges.

To be specific, there are six data or parameter processes and/or computational steps in the procedure, including:

1. Batching Strategy: In this data process, a shuffling algorithm is commonly applied [26]; firstly, each example $(\boldsymbol{x}, \boldsymbol{y})$ in training data $\mathcal{D}$ is randomly sorted to generate a list. Then, the batch set $\mathcal{B}(b_1, b_2, \ldots, b_{K-1}, b_K)$ is formed by scanning the list with a sliding window with the size of M, i.e., the batch size. Thus, K, the number of batches b in $\mathcal{B}$, equals $\lceil |\mathcal{D}|/M \rceil$. This data process sometimes may be performed more than once;
2. Parameter Initialization: In this parameter process, $\boldsymbol{\theta}_0$ are usually drawn randomly from a distribution (e.g., uniform) as the initialization parameters;
3. Loss Computation: Many kinds of deep multi-label classification models for automatic ICD coding can be applied in this step, including TextCNN, TextRNN and TextRCNN, which are compared in this paper. We uniformly note them as $f_{\boldsymbol{\theta}_k}(x_i)$. $f_{\boldsymbol{\theta}_k}(\boldsymbol{x}_i)$ means that the model $f(\boldsymbol{x}_i)$ takes the example $\boldsymbol{x}_i$ as the input with the kth iteration's parameter $\boldsymbol{\theta}_k$. The loss of the current kth iteration, $\mathcal{L}(\boldsymbol{\theta}_k)$, is the mean of the loss values obtained based on examples in $\mathcal{B}_k$. A loss value obtained based on one example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ in $\mathcal{B}_k$ is noted as $L_i(\boldsymbol{y}_i, f_{\boldsymbol{\theta}_k}(\boldsymbol{x}_i))$;

4. Gradient Update Estimation: According to $\boldsymbol{\theta}_k$, $f_{\boldsymbol{\theta}_k}(\boldsymbol{x}_i)$ and the examples in $\mathcal{B}_k$, the kth iteration's gradient updates, $\Delta\boldsymbol{\theta}_k$, can be estimated by the average of the gradients of the examples in $\mathcal{B}_k$, as shown in Figure 2;
5. Parameter Update: The $(k+1)$th iteration's parameter, $\boldsymbol{\theta}_{k+1}$, is calculated, based on $\boldsymbol{\theta}_k$ and $\Delta\boldsymbol{\theta}_k$ with the hyperparameter learning rate η;
6. Optimized Parameter Output: In general, the steps from 3 to 5 will loop E epochs until the model converges.

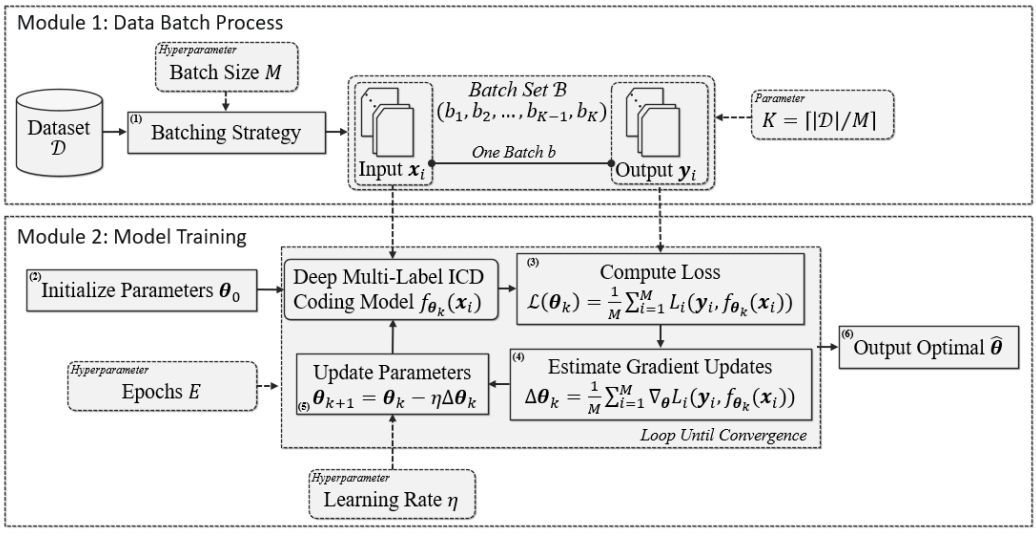

Figure 2. MBGD-based training procedure for the deep multi-label classification models for automatic ICD coding.

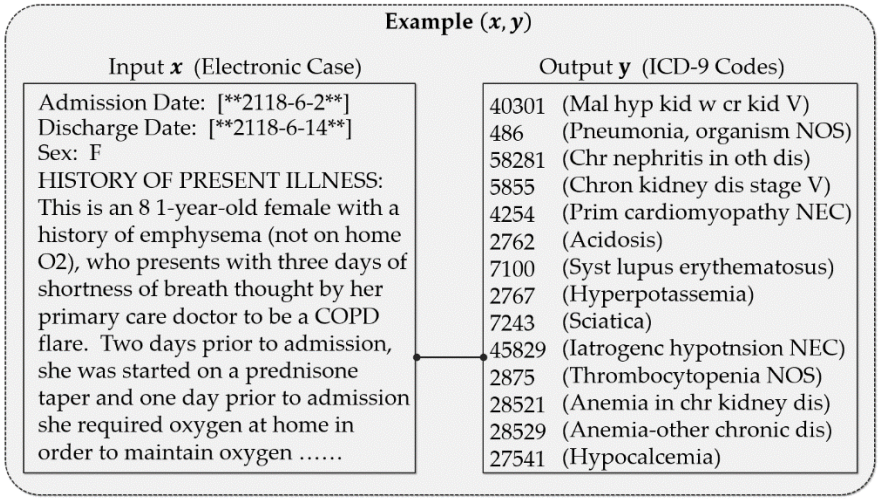

Figure 3. A sample of a real example $(\boldsymbol{x}, \boldsymbol{y})$.

Theoretically, $\Delta\boldsymbol{\theta}_k$ is the unbiased estimation of the kth iteration's gradient updates. Consequently, a larger batch size leads to a better gradient update approximation and results in a lower frequency of gradient updating. However, there is no "free lunch", in that it may also hurt generalization and result in finding poorer local optima [12].

Moreover, the commonly used batching strategy mentioned above makes each $\mathcal{B}_k$ carry different local statistics information from the global statistics information of $\mathcal{B}$, due to the inconsistency between the local data distribution of $\mathcal{B}_k$ and the global data distribution of $\mathcal{B}$. In addition, the extremely large label sets and imbalanced data distribution problems of automatic ICD coding also easily lead to a local overfitting on a small number of frequently occurring labels, due to the imbalanced label distribution of $\mathcal{D}$. To challenge these problems,

we propose a simple and effective curriculum batching strategy to conveniently replace the shuffling algorithm in the MBGD-based training procedure.

3.2. Curriculum-Batching Strategy

In the original MBGD-based training procedure for deep multi-label classification models for automatic ICD coding, $\mathcal{B}$ is generated by the batching strategy introduced in Section 3.1, and the generated $\mathcal{B}$ is used repeatedly by the following processes in the procedure and, sometimes, $\mathcal{B}$ may be regenerated. To challenge the problems mentioned earlier, we propose a curriculum batching strategy (CBS) data batch process module in order to generate batch sets $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$ with three types of data distribution, i.e., UDD, SDD and IDD, respectively. The structure of the new module is shown in Figure 4.

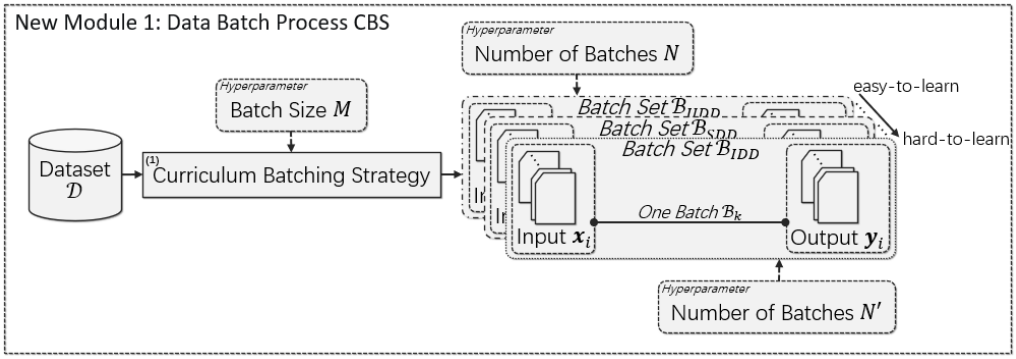

Figure 4. The curriculum batching strategy proposed in this paper has a similar structure to the batching strategy in the original MBGD-based training procedure for deep multi-label classification models for automatic ICD coding. Differently, $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$ will be iterated in turn from easy to hard.

The data batch process CBS can conveniently replace the original data batch process in Figure 2 (as depicted in Figure 4) except that the CBS will generate three batch sets, including $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$, which are sampled from the original training data, respectively, based on specific sampling algorithms. $\mathcal{B}_{UDD}$ is generated through a stratified sampling with replacement (SSR); $\mathcal{B}_{SDD}$ is generated based on the original shuffling algorithm; and $\mathcal{B}_{IDD}$ is generated by a probability sampling with replacement (PSR). The specific SSR and PSR methods will be described later; the original shuffling algorithm follows the batching strategy introduced in Section 3.1.

The intuitive assumption of the CBS is that $\mathcal{B}_{UDD}$ is easier to be learned by the deep multi-label ICD coding models than $\mathcal{B}_{SDD}$, and, in the same way, $\mathcal{B}_{SDD}$ is easier than $\mathcal{B}_{IDD}$. Moreover, $\mathcal{B}_{UDD}$ makes the models have an equally likely opportunity to learn the examples corresponding to each type of label, which can alleviate the imbalanced data distribution problem [29,30]. $\mathcal{B}_{SDD}$ allows models to learn every example corresponding to each type of label. $\mathcal{B}_{IDD}$ expects models to learn the ICD coding experience from examples through keeping the consistency between the local data distribution of $\mathcal{B}_k$ and the global data distribution of $\mathcal{B}$. So far, we have designed a curriculum strategy. According to the idea of curriculum learning [28,31], $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$ will be learned sequentially.

SSR for generating $\mathcal{B}_{UDD}$: The labels of the ICD coding system are diverse, extremely large and imbalanced. These issues make the examples in $\mathcal{D}$ have unequal opportunities to be learned by the models, which easily lead to a local overfitting on a small number of frequently occurring labels in a fixed number of loops.

Stratified sampling has the ability to ensure that every characteristic of the population is properly represented in one sample [32]. Consequently, we designed a simple SSR to generate a batch set satisfying the requirement that every example corresponding to each type of label in D has an equal chance to be learned by the models.

First, SSR samples M labels randomly, where M equals the batch size. Then for each sampled label, SSR samples one example with a replacement randomly from the examples in $\mathcal{D}$ whose corresponding labels contain the sampled label. Finally, the M examples constitute one batch. The above procedure will be repeated N times, where N is the size of the batch set $\mathcal{B}_{UDD}$. The sampled N batches form the batch set $\mathcal{B}_{UDD}$. In this paper, N is uniformly set to $\lceil |\mathcal{D}|/M \rceil$.

PSR for generating $\mathcal{B}_{IDD}$: Each $\mathcal{B}_k$ in $\mathcal{B}_{UDD}$ and $\mathcal{B}_{SDD}$ still carries different local statistics information from the global statistics information of D. In order to ensure the consistency between the local batch data distribution and the global training data distribution, we designed a simple PSR to generate a batch set satisfying the requirement that every example in $\mathcal{B}$ is drawn from the global data distribution of D with a replacement.

Differently from the SSR, the PSR firstly draws M labels according to the label distribution of $\mathcal{B}$, where M equals the batch size. Then, for each drawn label, the PSR samples one example randomly from the examples in $\mathcal{D}$ whose corresponding labels contain the drawn label. Finally, the sampled M examples constitute a batch. After repeating above procedure N' times, the batch set $\mathcal{B}_{IDD}$ is formed. In this paper, N', i.e., the size of the batch set $\mathcal{B}_{IDD}$, is uniformly set to $\lceil |\mathcal{D}|/M \rceil$, too.

4. Experiments

4.1. Experimental Dataset

In our experiments, the third version of the Medical Information Mart of Intensive Care (MIMIC-III) dataset [33] was adapted for evaluating the effectiveness of our proposed curriculum batching strategy for automatic ICD coding with multi-label classification models. As described in Section 3.1 above, we used one discharge summary (electronic case) of the NOTEEVENTS in MIMIC-III and its corresponding ICD-9 codes to form an example. Next, we conducted a series of data cleaning and preprocessing for all the samples to obtain the experimental dataset $\mathcal{D}$. The specific preprocesses were as follows:

- All the punctuation, numbers and stop words were removed from all the examples;
- The discharge summaries were segmented into tokens by using the space as the separator, and then we built a vocabulary $\mathcal{V}'$ based on these tokens;
- The TF-IDF values of each word in $\mathcal{V}'$ are calculated, based on all the examples, and only the top 10,000 words are kept to be used to build the final vocabulary $\mathcal{V}$.

After preprocessing, the basic information of $\mathcal{D}$ is listed in Table 2.

Table 2. The basic information of the MIMIC-III dataset after preprocessing in this paper.

$\lvert\mathcal{D}\rvert$	Unique Labels	Avg. Words per Example	Max Words in Examples
55,177	6918	898	4604
Min Words in Examples	**Avg. Labels per Example**	**Max Labels per Example**	**Min Labels per Example**
2	11	39	1

As shown in Table 2, $\mathcal{D}$ contains 55,177 different electronic medical records with 6918 different diseases in the ICD-9 code. Furthermore, in our experiments, $\mathcal{D}$ is further divided by a shuffling algorithm into a training dataset $\mathcal{D}_{train}$, a validation dataset $\mathcal{D}_{val}$ and a test dataset $\mathcal{D}_{test}$ in the ratio of 7:1:2.

4.2. Evaluation Metrics

In this paper, widely used micro-measures for multi-label classification tasks, including Precision$_{micro}$ (P$_{micro}$), Recall$_{micro}$ (R$_{micro}$), F1-Score$_{micro}$ (F1$_{micro}$) and F1-Score$_{macro}$ (F1$_{macro}$),

are used for evaluating the model performance, and the equations of these measures are shown as follows:

$$\mathrm{P}_{micro} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{1}$$

$$\mathrm{R}_{\mathrm{micro}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2}$$

$$\mathrm{F1}_{micro} = \frac{2 \times \mathrm{P}_{micro} \times \mathrm{R}_{micro}}{\mathrm{P}_{micro} + \mathrm{R}_{micro}} \tag{3}$$

$$\mathrm{P}_{macro} = \frac{1}{|L|} \sum_{l}^{|L|} P^{l}_{micro}, \ \mathrm{R}_{macro} = \frac{1}{|L|} \sum_{l}^{|L|} R^{l}_{micro} \tag{4}$$

$$\mathrm{F1}_{macro} = \frac{2 \times \mathrm{P}_{macro} \times \mathrm{R}_{macro}}{\mathrm{P}_{macro} + \mathrm{R}_{macro}} \tag{5}$$

where TP is the number of examples where the predictions are true positives; FP is the number of examples where the predictions are false positives; FN is the number of examples where the predictions are false negatives; and $|L|$ is the size of the label space.

4.3. Experimental Settings

In our experiments, we firstly trained TextCNN, TextRNN and TextRCNN for automatic ICD coding on $\mathcal{D}_{train}$ and tuned them on $\mathcal{D}_{val}$; three models were implemented, based on an open-source tool named NeuralNLP [34]. Finally, we applied the best model on $\mathcal{D}_{test}$ to get the results. A shuffling algorithm was applied for generating batches, and the batch size was set to different values to observe the sensitivity of the models. Adam was used as the optimizer, and the learning rate η and the epochs E were set to 0.008 and 150, respectively. The results are listed in Table 3.

Table 3. The automatic ICD coding performance of TextCNN, TextRNN and TextRCNN with different batch sizes. The best results are in bold.

Batch Size	TextCNN				TextRNN				TextRCNN			
	P_{micro}	R_{micro}	$\mathrm{F1}_{micro}$	$\mathrm{F1}_{macro}$	P_{micro}	R_{micro}	$\mathrm{F1}_{micro}$	$\mathrm{F1}_{macro}$	P_{micro}	R_{micro}	$\mathrm{F1}_{micro}$	$\mathrm{F1}_{macro}$
500	0.5365	0.2341	0.3254	0.0237	0.4725	0.2005	0.2815	0.0243	0.4867	0.2324	0.3146	0.0302
1000	0.5456	0.2311	0.3246	0.0222	0.5084	0.1814	0.2674	0.0192	0.5021	0.2265	0.3121	0.0264
2000	0.5397	0.2260	0.3186	0.0216	0.4769	0.1761	0.2572	0.0183	0.4945	0.2250	0.3092	0.0261
5000	0.5578	0.2207	0.3163	0.0197	0.4941	0.1644	0.2467	0.0162	0.4677	0.2379	0.3154	0.0265
7000	0.5403	0.2241	0.3168	0.0214	0.5541	0.1383	0.2214	0.0108	0.4694	0.2401	0.3174	0.0256

CBS was applied on TextCNN, TextRNN and TextRCNN for the automatic ICD coding through directly replacing the original shuffling algorithm-based batching strategy. We refer to them by "TextCNN+CBS", "TextRNN+CBS" and "TextRCNN+CBS". $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$ were built offline previously, and they were switched to be utilized in the training procedure every 50 epochs. For the sake of fairness, the other settings are the same as the settings described earlier, and the results are listed in Table 4.

Table 4. The automatic ICD coding performance of TextCNN+CBS, TextRNN+CBS and TextRCNN+CBS with different batch sizes. The best results are in bold.

Batch Size	TextCNN+CBS				TextRNN+CBS				TextRCNN+CBS			
	P_{micro}	R_{micro}	$\mathrm{F1}_{micro}$	$\mathrm{F1}_{macro}$	P_{micro}	R_{micro}	$\mathrm{F1}_{micro}$	$\mathrm{F1}_{macro}$	P_{micro}	R_{micro}	$\mathrm{F1}_{micro}$	$\mathrm{F1}_{macro}$
500	0.8121	0.6251	0.7064	0.5789	0.8072	0.4487	0.5766	0.4619	0.8246	0.6004	0.6949	0.5669
1000	0.8403	0.6693	0.7449	0.5802	0.7919	0.3605	0.4954	0.3300	0.8546	0.6167	0.7161	0.5563
2000	0.8507	0.5813	0.6907	0.4234	0.7226	0.2042	0.3184	0.0551	0.8304	0.5230	0.6418	0.4738
5000	0.8432	0.4940	0.6230	0.2898	0.6983	0.1510	0.2483	0.0198	0.8350	0.4142	0.5530	0.3602
7000	0.8005	0.3468	0.4839	0.0882	0.6638	0.1063	0.1832	0.0062	0.7800	0.3240	0.4580	0.1571

One ReLS method mentioned in [10] was applied directly to TextCNN, TextRNN and TextRCNN as a reference comparison model, and we named the results TextCNN+ReLS, TextRNN+ReLS and TextRCNN+ReLS. The settings of the models are the same as the settings mentioned earlier. Their results are listed in Table 5.

Table 5. The automatic ICD coding performance of TextCNN+ReLS, TextRNN+ReLS and TextRCNN+ReLS with different batch sizes. The best results are in bold.

Batch Size	TextCNN+ReLS				TextRNN+ReLS				TextRCNN+ReLS			
	P_{micro}	R_{micro}	$F1_{micro}$	$F1_{macro}$	P_{micro}	R_{micro}	$F1_{micro}$	$F1_{macro}$	P_{micro}	R_{micro}	$F1_{micro}$	$F1_{macro}$
500	0.4987	0.3412	0.4052	0.0977	0.4391	0.3107	0.3639	0.0816	0.4611	0.3471	0.3961	0.1030
1000	0.5091	0.3320	0.4019	0.0973	0.4171	0.3016	0.3501	0.0781	0.5065	0.3401	0.4069	0.0981
2000	0.5574	0.3474	0.4280	0.1010	0.4757	0.3052	0.3719	0.0817	0.5318	0.3624	0.4310	0.1067
5000	0.6290	0.3241	0.4277	0.0829	0.4882	0.3052	0.3756	0.0854	0.5638	0.3529	0.4341	0.1046
7000	0.6190	0.3310	0.4313	0.0824	0.5157	0.2890	0.3704	0.0813	0.5408	0.3567	0.4299	0.1068

Moreover, every experiment described above was run three times, and the average of the results is reported.

4.4. Overall Results

First of all, comparing $F1_{micro}$ in Table 3 with that in Table 4 shows clearly that, after changing the original batching strategy to CBS, the performance of TextCNN, TextRNN and TextRCNN improved by more than double. In particular, the $F1_{macro}$ increased from less than 5% to around 50% ($F1_{macro}$ is more affected by the long-tailed label). Moreover, the results shown in Tables 4 and 5 demonstrate that, with a simple and effective curriculum batching strategy substitution, the results of TextCNN+CBS, TextRNN+CBS and TextRCNN+CBS are much better than TextCNN+ReLS, TextRNN+ReLS and TextRCNN+ReLS, which use a complex label set reconstruction method. These results demonstrate that our proposed curriculum batching strategy for automatic ICD coding with multi-label classification models is effective.

All the results state that feeding the training data to the deep multi-label ICD coding model with an easy-to-learn data distribution first, and then, gradually, with a hard-to-learn data distribution can effectively improve the model's performance.

In order to illustrate the ability of our proposed strategy for automatic ICD coding with deep multi-label classification models without harming the generalization ability of the models, we also applied the models TextCNN+CBS, TextRNN+CBS and TextRCNN+CBS directly on $\mathcal{D}_{train}$. The results are listed in Table 6. By comparing the $F1_{micro}$ in Tables 4 and 6, it is easy to find that the performance of the models on the training data is comparable to that on the test data. It means that TextCNN+CBS, TextRNN-+CBS and TextRCNN+CBS do not suffer from the overfitting problem.

Table 6. The automatic ICD coding performance obtained by applying the models TextCNN+CBS, TextRNN+CBS and TextRCNN+CBS directly on the training data.

Batch Size	TextCNN+CBS (On Training Data)				TextRNN+CBS (On Training Data)				TextRCNN+CBS (On Training Data)			
	P_{micro}	R_{micro}	$F1_{micro}$	$F1_{macro}$	P_{micro}	R_{micro}	$F1_{micro}$	$F1_{macro}$	P_{micro}	R_{micro}	$F1_{micro}$	$F1_{macro}$
500	0.7934	0.5898	0.6766	0.8312	0.7886	0.4219	0.5496	0.6854	0.8039	0.5650	0.6636	0.8132
1000	0.8198	0.6353	0.7157	0.8179	0.7755	0.3395	0.4721	0.4451	0.8339	0.5805	0.6842	0.7935
2000	0.8290	0.5472	0.6592	0.5278	0.7195	0.1919	0.3029	0.0612	0.8076	0.4939	0.6130	0.6502
5000	0.8225	0.4658	0.5947	0.3245	0.6984	0.1400	0.2332	0.0208	0.8158	0.3892	0.5263	0.4777
7000	0.7839	0.3269	0.4614	0.0902	0.6672	0.0979	0.1707	0.0066	0.7908	0.2624	0.3938	0.1954

4.5. Detailed Analysis

It can be seen from the results listed in Tables 3–5 that the proposed curriculum batching strategy for automatic ICD coding with deep multi-label classification models improves the results of $F1_{micro}$ as well as the results of P_{micro} and R_{micro}. This result is different from the results obtained by the models TextCNN, TextRNN, TextRCNN, TextCNN+ReLS, TextRNN+ReLS and TextRCNN+ReLS. They improve the P_{micro} performance while sacrificing their R_{micro} performance. This further demonstrates that, after applying our proposed CBS to the training procedure, the generalization ability of the trained models has improved.

Our results are also consistent with the results of other existing research. Larger batch sizes result in a lower frequency of gradient updates and better gradient update approximation, but they hurt the performance (as shown in Tables 3–5). Therefore, the batch size should not be too large, and, of course, it should also not be too small. In our experiments, when the batch size is set to 1000, the results are the best for TextCNN+CBS and TextRCNN+CBS, and when it is set to 500, the results are the best for TextRNN+CBS.

Moreover, we plotted the training procedures for the nine models whose $F1_{micro}$ are shown in bold in Tables 3–5 in order to observe the convergence of every model. The training procedures are grouped according to their base models, i.e., TextCNN, TextRNN and TextRCNN, and presented in Figures 5–7, respectively. In the three figures, the X-axis is the number of epochs and the Y-axis is the loss value; the loss is calculated using BCEWITHLOGITSLOSS in PyTroch, and its calculation formula is as follows:

$$\ell(x, y) = [y \cdot log\sigma(x) + (1 - y) \cdot log\ (1 - \sigma(x))], \tag{6}$$

It can be seen in these figures that all the models converge after 150 epochs.

It can also be observed from Figures 5–7 that TextCNN+CBS, TextRNN+CBS and TextRCNN+CBS converge the fastest, respectively, in their group. TextCNN+ReLS, TextRNN+ReLS and TextRCNN-+ReLS have the slowest rate of convergence, but they have better $F1_{micro}$ than their base models, TextCNN, TextRNN and TextRCNN. The results show that our proposed curriculum batching strategy for automatic ICD coding with multi-label classification models can help the models to find their better local optima quickly through a gradual learning process without hurting their generalization ability.

Moreover, the loss values of TextCNN+CBS, TextRNN-+CBS and TextRCNN+CBS, in Figures 5–7 at the 50th epoch and the 100th epoch, all have two interesting change points showing a steep rise and then a steep drop. These change points occur when $\mathcal{B}_{UDD}$ switches to $\mathcal{B}_{SDD}$, and $\mathcal{B}_{SDD}$ switches to $\mathcal{B}_{IDD}$, respectively. That means when the local data distribution changes dramatically, the loss will be affected. In other words, the performance of the models would be hurt at those points. However, owing to our proposed curriculum batching strategy, the models can quickly adapt to the new data distributions. It not only keeps good prediction performance, but also avoids the risk of falling into a local optimal solution.

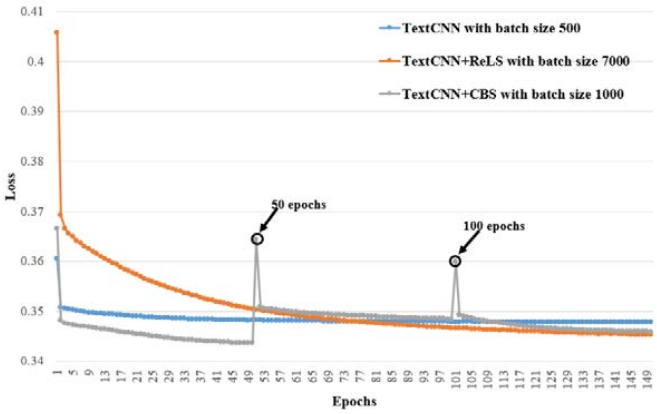

Figure 5. Training procedures for TextCNN-based models with different batch sizes whose $F1_{micro}$ are in bold in Tables 3–5.

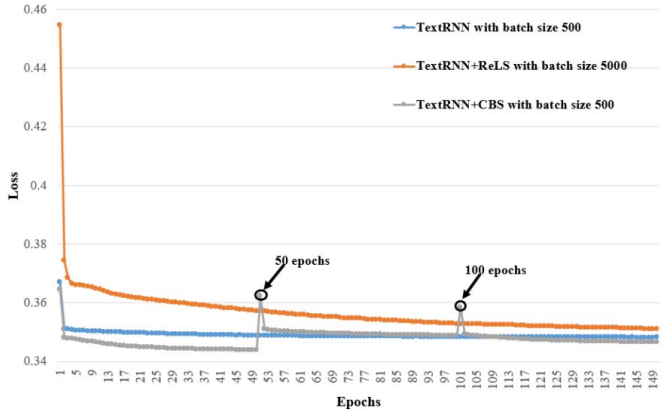

Figure 6. Training procedures for TextRNN-based models with different batch sizes whose $F1_{micro}$ are in bold in Tables 3–5.

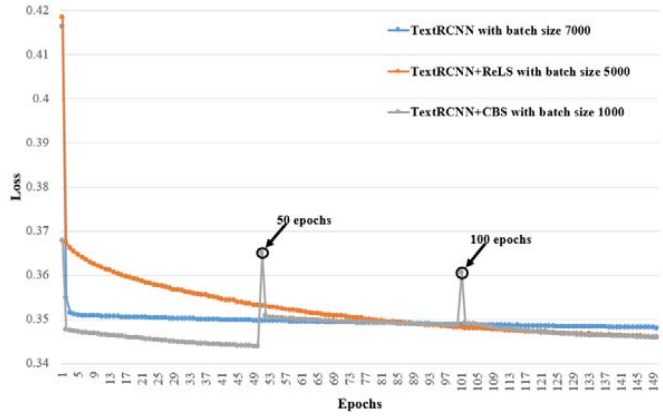

Figure 7. Training procedures for TextRCNN-based models with different batch sizes whose $F1_{micro}$ are in bold in Tables 3–5.

4.6. Evaluating the Learning Difficulties of $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$

In this paper, the proposed strategy is based on an intuitive assumption that $\mathcal{B}_{UDD}$ is easier to be learned by the deep multi-label ICD coding models than $\mathcal{B}_{SDD}$, and $\mathcal{B}_{SDD}$ is easier to be learned than $\mathcal{B}_{IDD}$. The implication of this assumption is that the data distribution of $\mathcal{B}_{IDD}$ is more like the data distribution of $\mathcal{D}$ than that of $\mathcal{B}_{SDD}$, and the data distribution of $\mathcal{B}_{SDD}$ is more like that of $\mathcal{D}$ than that of $\mathcal{B}_{UDD}$.

In order to further verify the effectiveness of the proposed curriculum batching strategy, the data distribution similarities between the three generated batch sets and $\mathcal{D}$, i.e., the learning difficulties of $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$, are evaluated by using the Kullback–Leibler Divergence (KLD) and the Jensen–Shannon Distance (JSD). The evaluation results are listed in Table 7. It can clearly see that the data distribution of $\mathcal{B}_{UDD}$ is the least like that of $\mathcal{D}$. $\mathcal{B}_{SDD}$ is the next, and the data distribution of $\mathcal{B}_{IDD}$ is the most like that of $\mathcal{D}$.

Table 7. Learning difficulties of $\mathcal{B}_{UDD}$, $\mathcal{B}_{SDD}$ and $\mathcal{B}_{IDD}$ evaluated by KLD and JSD, respectively. The smaller the KLD value is, the more difficult to be learned the batch set is. The smaller the JSD value is, the less difficult to be learned the batch set is.

Method	$\mathcal{B}_{UDD}$	$\mathcal{B}_{SDD}$	$\mathcal{B}_{IDD}$
KLD	0.124	0.103	0.090
JSD	33.3×10^{-3}	12.8×10^{-3}	8.97×10^{-3}

4.7. Ablation Experiments of CBS

To prove the effectiveness of the curriculum in the CBS, we designed and implemented ablation experiments for the three models, and the experimental results are shown in Table 8. It can be clearly observed that all the performance indicators of the three models are steadily improving with the increase in the curricula.

Table 8. Results of ablation experiment (batch size = 1000).

Model	Curriculum	P_{micro}	R_{micro}	$F1_{micro}$	$F1_{macro}$
TextCNN	$\mathcal{B}_{SDD}$	0.5456	0.2311	0.3246	0.0222
	$\mathcal{B}_{UDD}$+$\mathcal{B}_{SDD}$	0.7556	0.3224	0.4407	0.1584
	$\mathcal{B}_{UDD}$+$\mathcal{B}_{SDD}$+$\mathcal{B}_{IDD}$	0.8403	0.6693	0.7449	0.5802
TexRNN	$\mathcal{B}_{SDD}$	0.5084	0.1814	0.2674	0.0192
	$\mathcal{B}_{UDD}$+$\mathcal{B}_{SDD}$	0.7963	0.3832	0.5174	0.2536
	$\mathcal{B}_{UDD}$+$\mathcal{B}_{SDD}$+$\mathcal{B}_{IDD}$	0.7919	0.3605	0.4954	0.3300
TextRCNN	$\mathcal{B}_{SDD}$	0.5021	0.2265	0.3121	0.0264
	$\mathcal{B}_{UDD}$+$\mathcal{B}_{SDD}$	0.7378	0.2590	0.3766	0.1399
	$\mathcal{B}_{UDD}$+$\mathcal{B}_{SDD}$+$\mathcal{B}_{IDD}$	0.8546	0.6167	0.7161	0.5563

4.8. Evaluating the Long-Tailed Label Learning Performance

The improvements in the automatic ICD coding performance benefit from our proposed curriculum batching strategy. The improvements are also related to the ability of the proposed strategy to promote the learning of the long-tailed labels (labels with a low frequency in $\mathcal{D}$, as shown in Figure 8). In the early phase of the training, we used $\mathcal{B}_{UDD}$, which satisfies the uniform distribution, to train the models. This not only provided a simple learning task, but also allowed the information in the long-tailed labels in the MIMIC-III dataset to be learned fully. To evaluate this, we set a threshold $\gamma = 219$. If the frequencies of labels were lower than γ, we considered these labels the long-tailed labels. Then, we calculated the precisions and recalls of the long-tailed labels of the models, and the results are listed in Table 9. It vividly shows that the proposed curriculum batching strategy improves the long-tailed label learning ability of deep multi-label classification models for automatic ICD coding.

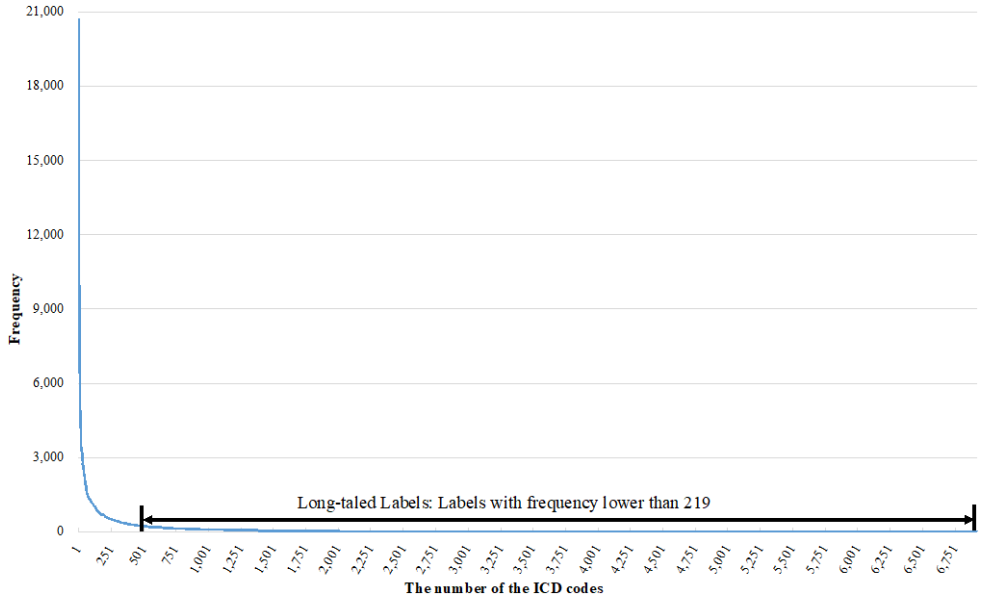

Figure 8. Long-tailed labels in the MIMIC-III dataset and our definition in this paper.

Table 9. Comparing the recalls of the long-tailed labels of TextCNN, TextRNN and TextRCNN with their CBS-improved models for automatic ICD coding.

Model	Precision	Recall
TextCNN	0.0188	0.0498
TextRNN	0.0169	0.0442
TextRCNN	0.0103	0.0269
TextCNN+CBS	0.2599	0.6369
TextRNN+CBS	0.2593	0.6617
TextRCNN+CBS	0.1943	0.4778

5. Conclusions and Future Work

In this paper, we observed that extremely large ICD coding label sets and imbalanced label distribution lead to inconsistency between the local batch data distribution and the global training data distribution, and this inconsistency brings the overfitting problem into the mini-batch gradient descent algorithm-based training procedure for deep multi-label classification models for automatic ICD coding. Therefore, we proposed and investigated a simple and effective curriculum batching strategy to replace the original shuffling algorithm-based batching strategy in the training procedure. The proposed strategy emphasizes that the model training procedure should be gradual, rather than random; the proposed curriculum batching strategy realizes a gradual model training procedure by generating three batch sets offline through applying three specific sampling algorithms. These batch sets satisfy the uniform data distribution, the shuffling data distribution and the original training data distribution, respectively. The learning tasks corresponding to these batch sets range from easy to hard. A series of experiments demonstrate the effectiveness of the proposed batching strategy. After replacing the original shuffling algorithm-based batching strategy with the proposed curriculum batching strategy, the performance of the three investigated deep multi-label classification models for automatic ICD coding all improve dramatically. At the same time, the models avoid the overfitting issue and all show better ability to learn the long-tailed label information. In addition, we also compared the performance with a state-of-the-art label set reconstruction model for automatic ICD coding. The performance of ours is consistently higher than that of the state-of-the-art model. These results further prove the effectiveness of the proposed curriculum batching strategy for automatic ICD coding with multi-label classify-cation models.

Possible directions for future work may include choosing the course of the CBS strategy adaptively according to the convergence of the multi-label classification model during the training. Currently, the order of courses and the number of times they are used are fixed. Another direction is to combine transfer learning and pre-train the model with data sets in the medical field.

Author Contributions: Conceptualization, Y.W.; methodology, Y.W.; software, X.H. (Xu Han); validation, X.H. (Xu Han) and X.H. (Xuechao Hao); formal analysis, Y.W.; investigation, Y.W.; resources, Y.W., X.H. (Xuechao Hao) and H.S.; data curation, Y.W. and X.H. (Xu Han); writing—original draft preparation, Y.W. and X.H. (Xu Han); writing—review and editing, Y.W., X.H. (Xuechao Hao) and X.H. (Xu Han); visualization, Y.W., X.H. (Xuechao Hao) and X.H. (Xu Han); supervision, Y.W., X.H. (Xuechao Hao), T.Z. and H.S.; project administration, Y.W. and X.H. (Xuechao Hao). All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Scientific and Technology Innovation Capability Enhancement Program, Chengdu University of Information Technology (No. KYQN202209); the 1·3·5 project for disciplines of excellence, West China Hospital, Sichuan University (No. ZYJC18010).

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki, and approved by the Ethics Committee of West China Hospital of Sichuan University (2019-473), and registered at www.chictr.org.cn, accessed on 16 September 2022 (ChiCTR1900025160).

Informed Consent Statement: Informed consent was obtained from all subjects involved in this study.

Data Availability Statement: The dataset used in this study is from a large-scale freely accessible critical care database—MIMIC-III, which is widely used in ICD coding tasks. MIMIC-III ('Medical Information Mart for Intensive Care') is a large, single-center database comprising information relating to patients admitted to critical care units at a large tertiary-care hospital. The data include vital signs, medications, laboratory measurements, observations and notes charted by care providers, fluid balance, procedure codes, diagnostic codes, imaging reports, hospital length of stay, survival data, and more. This dataset is published in the paper by Johnson A E W, Pollard T J, Shen L, et al., *MIMIC-III, a freely accessible critical care database*[J]. MIMIC-III is provided as a collection of commas separated value (CSV) files, along with scripts to help with importing the data into database systems including PostreSQL, MySQL, and MonetDB. Researchers can request access via a process documented on the MIMIC website (http://mimic.physionet.org) (accessed on 16 September 2022). There are two key steps that must be completed before access is granted: the researcher must complete a recognized course in protecting human research participants that includes Health Insurance Portability and Accountability Act (HIPAA) requirements; and the researcher must sign a data use agreement, which outlines appropriate data usage and security standards and forbids efforts to identify individual patients. Approval requires at least a week. Once an application has been approved, the researcher will receive emails containing instructions for downloading the database from PhysioNetWorks, a restricted access component of PhysioNet.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. World Health Organization. *ICD-10: International Statistical Classification of Diseases and Related Health Problems: Tenth Revision. Volume 1: Tabular List*; World Health Organization: Geneva, Switzerland, 2004.
2. Yu, Y.; Li, M.; Liu, L.; Fei, Z.; Wu, F.-X.; Wang, J. Automatic ICD code assignment of Chinese clinical notes based on multilayer attention BiRNN. *J. Biomed. Inform.* **2019**, *91*, 103114. [CrossRef] [PubMed]
3. Nguyen, A.N.; Truran, D.; Kemp, M.; Koopman, B.; Conlan, D.; O'Dwyer, J.; Zhang, M.; Karimi, S.; Hassanzadeh, H.; Lawley, M.J.; et al. Computer-Assisted Diagnostic Coding: Effectiveness of an NLP-Based Approach Using SNOMED CT to ICD-10 Mappings. In Proceedings of the AMIA Annual Symposium Proceedings (AMIA 2018), San Francisco, CA, USA, 3–7 November 2018; pp. 807–816.
4. O'Malley, K.J.; Cook, K.F.; Price, M.D.; Wildes, K.R.; Hurdle, J.F.; Ashton, C.M. Measuring Diagnoses: ICD Code Accuracy. *Health Serv. Res.* **2005**, *40*, 1620–1639. [CrossRef] [PubMed]
5. Rotmensch, M.; Halpern, Y.; Tlimat, A.; Horng, S.; Sontag, D. Learning a Health Knowledge Graph from Electronic Medical Records. *Sci. Rep.* **2017**, *7*, 5994. [CrossRef] [PubMed]
6. Stanfill, M.H.; Williams, M.; Fenton, S.H.; Jenders, R.A.; Hersh, W. A systematic literature review of automated clinical coding and classification systems. *J. Am. Med. Inform. Assoc.* **2010**, *17*, 646–651. [CrossRef] [PubMed]
7. Kaur, R.; Ginige, J.A.; Obst, O. A Systematic Literature Review of Automated ICD Coding and Classification Systems Using Discharge Summaries. *arXiv* **2021**, arXiv:2107.10652v1.
8. Blanco, A.; Perez, A.; Casillas, A. Extreme Multi-Label ICD Classification: Sensitivity to Hospital Service and Time. *IEEE Access* **2020**, *8*, 183534–183545. [CrossRef]
9. Vu, T.; Nguyen, D.Q.; Nguyen, A. A Label Attention Model for ICD Coding from Clinical Text. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20), Yokohama, Japan, 7–15 January 2021; pp. 3335–3341.
10. Blanco, A.; de Viñaspre, O.P.; Pérez, A.; Casillas, A. Boosting ICD multi-label classification of health records with contextual embeddings and label-granularity. *Comput. Methods Programs Biomed.* **2020**, *188*, 105264. [CrossRef] [PubMed]
11. Song, C.; Zhang, S.; Sadoughi, N.; Xie, P.; Xing, E. Generalized Zero-Shot Text Classification for ICD Coding. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-PRICAI 2020), Yokohama, Japan, 7–15 January 2021; pp. 4018–4024.
12. Smith, S.L.; Kindermans, P.-J.; Ying, C.; Le, Q.V. Don't Decay the Learning Rate, Increase the Batch Size. In Proceedings of the Sixth International Conference on Learning Representations (ICLR 2018), Vacounver, Canada, 30 April–3 May 2018; pp. 1–11.
13. Wu, Y.; Johnson, J. Rethinking "Batch" in BatchNorm. *arXiv* **2021**, arXiv:2105.07576v1.
14. de Lima, L.R.S.; Laender, A.H.F.; Ribeiro-Neto, B.A. A Hierarchical Approach to the Automatic Categorization of Medical Documents. In Proceedings of the Seventh International Conference on Information and Knowledge Management (CIKM 1998), Bethesda, MD, USA, 2–7 November 1998; pp. 132–139.
15. Perotte, A.; Pivovarov, R.; Natarajan, K.; Weiskopf, N.; Wood, F.; Elhadad, N. Diagnosis Code Assignment: Models and Evaluation Metrics. *J. Am. Med. Inform. Assoc.* **2014**, *21*, 231–237. [CrossRef] [PubMed]
16. Li, F.; Yu, H. ICD Coding from Clinical Text Using Multi-Filter Residual Convolutional Neural Network. *Proc. Conf. AAAI Artif. Intell.* **2020**, *34*, 8180–8187. [CrossRef] [PubMed]

17. Xie, X.; Xiong, Y.; Yu, P.S.; Zhu, Y. EHR Coding with Multi-scale Feature Attention and Structured Knowledge Graph Propagation. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM 2019), Beijing, China, 3–7 November 2019; pp. 649–658.
18. Mullenbach, J.; Wiegreffe, S.; Duke, J.; Sun, J.; Eisenstein, J. Explainable prediction of medical codes from clinical text. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2018), New Orleans, Louisiana, 1-6 June 2018; pp. 1101–1111.
19. Sadoughi, N.; Finley, G.P.; Fone, J.; Murali, V.; Korenevski, M.; Baryshnikov, S.; Axtmann, N.; Miller, M.; Suendermann-Oeft, D. Medical code prediction with multi-view convolution and description-regularized label-dependent attention. *arXiv* **2018**, *preprint*. arXiv:1811.01468.
20. Bhutto, S.R.; Wu, Y.; Yu, Y.; Jalbani, A.H.; Li, M. A Hybrid Pooling Based Deep Learning Framework for Automated ICD Coding. In Proceedings of the 2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Online, 9–12 December 2021; pp. 823–828.
21. Ruder, S. An Overview of Gradient Descent Optimization Algorithms. *arXiv* **2017**, arXiv:1609.04747v2.
22. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; The MIT Press: Cambridge, MA, USA, 2016.
23. Lin, T.; Kong, L.; Stich, S.U.; Jaggi, M. Extrapolation for Large-batch Training in Deep Learning. In Proceedings of the 37th International Conference on Machine Learning (ICML 2020), Vienna, Austria, 12–18 July 2020; pp. 6094–6104.
24. Masters, D.; Luschi, C. Revisiting Small Batch Training for Deep Neural Networks. *arXiv* **2018**, arXiv:1804.07612v1.
25. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning (ICML 2015), Lille, France, 6–11 July 2015; pp. 448–456.
26. Montavon, G.; Orr, G.B.; Müller, K.-R. *Neural Networks: Tricks of the Trade*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2012.
27. Elman, J.L. Learning and development in neural networks: The importance of starting small. *Cognition* **1993**, *48*, 71–99. [CrossRef] [PubMed]
28. Bengio, Y.; Louradour, J.; Collobert, R.; Weston, J. Curriculum Learning. In Proceedings of the 26th Annual International Conference on Machine Learning (ICML 2009), Montreal, QB, Canada, 14–18 June 2009; pp. 41–48.
29. Branco, P.; Torgo, L.; Ribeiro, R.P. A Survey of Predictive Modeling on Imbalanced Domains. *ACM Comput. Surv.* **2016**, *49*, 31. [CrossRef]
30. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
31. Krysi´nska, I.; Morzy, M.; Kajdanowicz, T. Curriculum Learning Revis-ited: Incremental Batch Learning with Instance Typicality Ranking. In Proceedings of the 30th International Conference on Artificial Neural Networks (ICANN 2021), Bratislava, Slovakia, 14—17 September 2021; pp. 279–291.
32. Liu, W.; Qian, H.; Zhang, C.; Shen, Z.; Xie, J.; Zheng, N. Accelerating Stratified Sampling SGD by Reconstructing Strata. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI 2020), Yokohama, Japan, 7–15 January 2021; pp. 2725–2731.
33. Johnson, A.E.W.; Pollard, T.J.; Shen, L.; Lehman, L.H.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Celi, L.A.; Mark, R.G. MIMIC-III, a freely accessible critical care database. *Sci. Data* **2016**, *3*, 160035. [CrossRef] [PubMed]
34. Liu, L.; Mu, F.; Li, P.; Mu, X.; Tang, J.; Ai, X.; Fu, R.; Wang, L.; Zhou, X. NeuralClassifier: An open-source neural hierarchical multi-label text classification toolkit. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Florence, Italy, 2 July–2 August 2019; pp. 87–92.