Effective Convolutional Attention Network for Multi-label Clinical Document Classification

Yang Liu[1], Hua Cheng[1], Russell Klopfer[1], Thomas Schaaf[1], Matthew R. Gormley[2]
[1]3M Health Information Systems, [2]Carnegie Mellon University
tomdzh1@gmail.com, {hcheng, rklopfer, tschaaf}@mmm.com, mgormley@cs.cmu.edu

Abstract

Multi-**l**abel **d**ocument **c**lassification (MLDC) problems can be challenging, especially for long documents with a large label set and a long-tail distribution over labels. In this paper, we present an effective convolutional attention network for the MLDC problem with a focus on medical code prediction from clinical documents. Our innovations are three-fold: (1) we utilize a deep convolution-based encoder with the squeeze-and-excitation networks and residual networks to aggregate the information across the document and learn meaningful document representations that cover different ranges of texts; (2) we explore multi-layer and sum-pooling attention to extract the most informative features from these multi-scale representations; (3) we combine binary cross entropy loss and focal loss to improve performance for rare labels. We focus our evaluation study on MIMIC-III, a widely used dataset in the medical domain. Our models outperform prior work on medical coding and achieve new state-of-the-art results on multiple metrics. We also demonstrate the language independent nature of our approach by applying it to two non-English datasets. Our model outperforms prior best model and a multilingual Transformer model by a substantial margin.

1 Introduction

In multi-label document classification (MLDC), we have a set of labeled data $\{\mathbb{X}, \mathbb{Y}\}$, $\mathbb{X} \in \mathbb{R}^{N \times D}$ and $\mathbb{Y} \in \mathbb{R}^{N \times L}$, where N is the number of documents, D is the feature dimension size of each document and L is the total number of labels. The i^{th} row of $\mathbb{Y}$ is a multi-hot vector representing the set of labels associated with the i^{th} document. The task is to learn a mapping between $\mathbb{X}$ and $\mathbb{Y}$ so that the labels of each document are predicted correctly.

MLDC has a great number of practical applications, one of which is automatic medical coding, where a patient encounter containing multiple records are assigned with appropriate medical codes. A large number of medical encounters need to be coded for billing purposes everyday. Professional clinical coders often use rule-based or simple ML-based systems to assign billing codes, but the large code space (viz. the ICD-10 code system with over 90,000 codes) and long documents are challenging for ML models. In addition, coding requires extracting useful information from specific locations across *the entire encounter* to support the assigned codes. Consequently, effective models with the capability of handling these challenges will have an immense impact in the medical domain by helping to reduce coding cost, improve coding accuracy and increase customer satisfaction.

Deep learning methods have been demonstrated to produce the state-of-the-art outcomes on benchmark MLDC and medical coding tasks (You et al., 2019a; Mullenbach et al., 2018; Chang et al., 2020), but demands remain for more effective and accurate solutions. In this paper, we propose **EffectiveCAN**, an **effective c**onvolution **a**ttention **n**etwork for MLDC. Our models try to strike a careful balance of simplicity and effective over-parameterization such that we can effectively model long documents and capture nuanced aspects of the whole document texts. Such a model is particularly useful for addressing the challenges of automatic medical coding. We evaluate our models on the widely used MIMIC-III dataset (Johnson et al., 2016), and attain state-of-the-art results across multiple commonly used metrics. We also demonstrate the language-independent nature of our approach by coding on two non-English datasets. Our model outperforms prior best model and a multilingual transformer model by a substantial margin.

2 Related Works

Previous deep learning methods for MLDC involve various neural network architectures to learn the semantic embeddings of the document texts. For example, XML-CNN proposed by Liu et al.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 5941–5953
November 7–11, 2021. ©2021 Association for Computational Linguistics

(2017) employs a 1-dimension convolutional network along with dynamic pooling to learn the text representation. RNN-based sequence-to-sequence models, such as SGM (Yang et al., 2018) and SU4MLC (Lin et al., 2018) use an encoder to encode the information of the input text and a decoder to generate the predicted labels. AttentionXML proposed by You et al. (2019a) leverages the BiLSTM and label-aware attention to capture the most relevant texts for each label. As a follow-up, MAGNET (Pal et al., 2020) incorporates graph neural network to capture the attentive dependency structure among the labels. More recently, transformers such as the X-transformer (Chang et al., 2020) have also been introduced. X-transformer tackles MLDC in three steps: label clustering, transformer classification and label ranking.

There is a surge in neural network models employed for automatic medical coding in the past several years. In particular, recent works have utilized attention mechanism to improve automatic coding performance. Shi et al. (2017) applied LSTMs to produce represtations of the discharge summary and used attention to predict the top 50 codes. Mullenbach et al. (2018) proposed CAML that applies separate attention for each label, which generates better label-specific representations for label prediction. They also used the label descriptions to regularize the model (called DL-CAML) in an attempt to improve the prediction of rare labels. To improve the classification performance, Xie et al. (2019) used the multi-scale convolutional attention while Li and Yu (2020) employed multi-filter convolution to learn text patterns of different lengths. Furthermore, to incorporate the inner relationship of the labels, HyperCore (Cao et al., 2020) integrated a hyperbolic representation learning method and a graph convolutional network, and Lu et al. (2020) utilized multi-graph knowledge aggregation. Vu et al. (2020) proposed to combine Bi-LSTM and an extension of structured self-attention mechanism for ICD code prediction.

3 EffectiveCAN Model

In this section, we introduce our EffectiveCAN model (Figure 1), which is composed of four major components: an input layer that transforms the raw document texts into pretrained word embeddings, a deep convolution-based encoder that combines the information of adjacent words and learns meaningful representations of the document texts, an attention component that selects the most important text features and generates label-specific representations for each label, and an output layer that produces the final predictions.

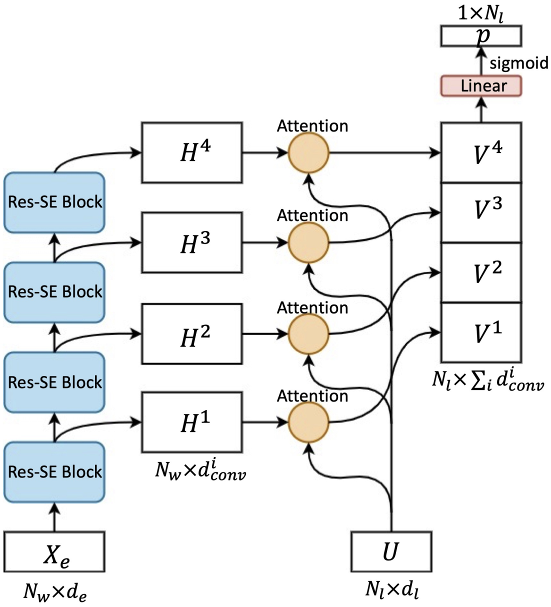

Figure 1: The architecture of EffectiveCAN.

The model structure is primarily designed for generating better predictions on multi-label classification tasks from three aspects: (1) generating meaningful representations for input texts; (2) selecting informative features from text representations for label prediction; (3) preventing overconfidence on frequent labels. Firstly, in order to obtain high-quality representations of the document texts, we incorporate the **s**queeze-and-**e**xcitation (SE) network and the residual network into the convolution-based encoder. The encoder consists of multiple encoding blocks to enlarge the receptive field and capture text patterns with different lengths. Secondly, instead of only using the last encoder layer output for attention, we extract all encoding layer outputs and apply the attention to select the most informative features for each label. Finally, to cope with the long-tail distribution of the labels, we use a combination of the binary cross entropy loss and focal loss to make the model perform well on both frequent and rare labels.

3.1 Input Layer

Our model takes a word sequence as the input and each word is transferred to a word embedding of size d_e. Assuming the document has N_w number of words, the input will be a word embedding matrix

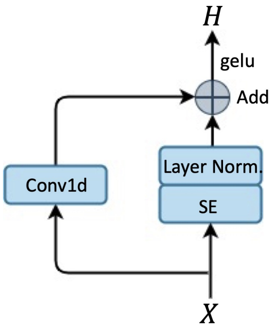

Figure 2: The structure of a Res-SE block containing a SE module and a residual module.

$X_e = [x_1, \ldots, x_{N_w}] \in \mathbb{R}^{N_w \times d_e}$.

3.2 Convolutional Encoder

To transform the document into informative representations, the input word embeddings X_e first go through a convolution-based encoder that consists of multiple **res**idual **s**queeze-and-**e**xcitation convolutional blocks (Res-SE blocks). Each Res-SE block, as shown in Figure 2, is composed of two parallel modules that are referred to as the SE module and the residual module.

In recent years, transformer-based models with self-attention modules have shown to be effective in text classification tasks (Devlin et al., 2018; Liu et al., 2019). However, for our applications we use a convolutional encoder instead of a self-attention one for two reasons: (1) ICD code predictions are often associated with a span of texts in the input. Convolutional operations can effectively aggregate the information of text spans and output meaningful representations for downstream predictions; (2) Clinical documents are usually long (i.e. MIMIC-III document have an average of 1500 words). A convolutional encoder is more time and space efficient than a self-attention encoder for modeling long documents.

3.2.1 SE Module

The SE module contains a squeeze-and-excitation network (Hu et al., 2018) followed by layer normalization (Ba et al., 2016). The SE network can adaptively adjust the weighting of each feature map and refine the convolutional features. Here we use the SE network to enhance the learning of document representations for the down-stream prediction task. The structure of the SE network in our model is shown in Figure 3. In this network, we first apply a standard 1-dimensional convolutional layer on the input to aggregate the information of adjacent word embeddings. Suppose the convolutional filter applied on the input matrix X is $W_c \in \mathbb{R}^{k \times d_e \times d_{conv}}$, where k is the filter size, d_e is the in-channel size (the size of input embedding) and d_{conv} is the out-channel size (the size of output embedding). The 1-dimensional convolution is computed as

$$c_i = W_c * x_{i:i-k+1} + b_c \qquad (1)$$

where $*$ is the convolution operator and b_c the bias. The output convolutional features can be represented as $C = [c_1, \ldots, c_{N_w}]$ with $C \in \mathbb{R}^{N_w \times d_{conv}}$.

The SE network then uses a two-stage process, 'squeeze' and 'excitation', to compute the channel-dependent coefficients to enhance the convolutional features. In the 'squeeze' stage, each channel is compressed into a single numeric value via global average pooling: $z_c = GAP(C)$. Here $z_c \in \mathbb{R}^{d_{conv}}$ can be treated as a channel descriptor that aggregates the global spatial information of C. In the 'excitation' stage, the channel descriptor goes through a dimensionality-reduction-layer with reduction ratio r followed by a dimensionality-increasing-layer back to the channel dimension of C. The reduction ratio r is a tunable parameter and we use $r = 20$ in our model. The excitation step can be written as

$$s_c = sigmoid(W_{fc2} \cdot relu(W_{fc1} \cdot z_c + b_{fc1}) + b_{fc2}) \qquad (2)$$

where $W_{fc1} \in \mathbb{R}^{\frac{d_{conv}}{r} \times d_{conv}}$, $b_{fc1} \in \mathbb{R}^{\frac{d_{conv}}{r}}$, $W_{fc2} \in \mathbb{R}^{d_{conv} \times \frac{d_{conv}}{r}}$ and $b_{fc2} \in \mathbb{R}^{d_{conv}}$ are the weights and biases of the fully-connected linear layers. Next, we rescale the convolutional feature C with s_c by: $\widetilde{X} = scale(C, s_c)$, where $scale$ denotes the channel-wise multiplication between C and s_c.

Eventually, $\widetilde{X}$ is normalized and used as the output of the SE module. In particular, we employ the layer normalization (Ba et al., 2016) that's widely used for stabilizing the hidden layer distribution and smoothing the gradients in NLP tasks (Devlin et al., 2018; Hou et al., 2019).

3.2.2 Residual Module

In addition to the SE module, we also simultaneously transform input X and add it to the SE module output as in the residual network (He et al., 2016), which reduces the gradient vanishing issue in the deep encoder structure. In order to avoid dimension mismatch, we transform the input X

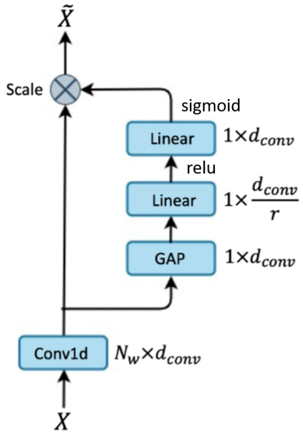

Figure 3: The architecture of the SE network.

into X' by using a filter-size-1 convolutional layer. Then we add X' with the SE module output $\widetilde{X}$.

Finally, we apply the $gelu$ activation function to generate the output of the Res-SE block:

$$H = gelu(\widetilde{X} + X') \tag{3}$$

3.3 Attention

We use the label-wise attention (Mullenbach et al., 2018) to generate label specific representations from H. Since our convolutional encoder contains multiple Res-SE blocks that generate multi-scale representations of the document texts, we perform multi-layer attention, which attends to outputs of all Res-SE blocks. In this way, each label is allowed to select the most relevant features from a rich feature space extracted by the encoder. Assuming $U \in \mathbb{R}^{N_l \times d_l}$ represents the label embedding matrix, where N_l is the number of labels and d_l the embedding size of each label. To attend to the i^{th} Res-SE layer output $H^i \in \mathbb{R}^{N_w \times d^i_{conv}}$, U is first mapped to $U' \in \mathbb{R}^{N_l \times d^i_{conv}}$ via a filter-size-1 convolutional layer to avoid dimension mismatch. The attention weights are then computed by

$$A^i = softmax(U' \cdot H^{i^T}) \tag{4}$$

Here, each of the j^{th} column of $A^i \in \mathbb{R}^{N_l \times N_w}$ is a weight vector measuring how informative the text representations in H are for the j^{th} label. Next, we generate the label specific representations: $V^i = A^i \cdot H^i$, where the j^{th} column in $V^i \in \mathbb{R}^{N_l \times d^i_{conv}}$ is the label specific representation for the j^{th} label, generated from the i^{th} Res-SE layer output.

We repeat the attention process for all Res-SE layer outputs, then concatenate the label specific representations:

$$V = concat(V^1, \dots, V^{N_{Res-SE}}) \tag{5}$$

where N_{Res-SE} is the number of Res-SE blocks. The resulted $V \in \mathbb{R}^{N_l \times \sum_i d^i_{conv}}$ will be used for the final prediction.

When the application domain has a large label space but insufficient data points, a multi-layer attention model can be difficult to train, especially for deep networks. Therefore we also experiment with sum-pooling attention where we first transform each convolutional layer to have the same dimention as the last layer, then sum all the layers and apply attention to the summed output. The resulting $V' \in \mathbb{R}^{N_l \times d^{last-layer}_{conv}}$ is used for the final prediction.

3.4 Output Layer

After obtaining the label specific representations, we compute the probability for each label by using a fully connected layer followed by a sum-pooling operation and a $sigmoid$ transformation:

$$p = sigmoid(pooling(W_{fc} \cdot V^T + b_{fc})) \tag{6}$$

where $W_{fc} \in \mathbb{R}^{\sum_i d^i_{conv} \times N_l}$ and $b_{fc} \in \mathbb{R}^{N_l}$. Here, the j^{th} value in p is the predicted probability for the j^{th} label to be present given the document texts.

3.5 Loss Function

Binary cross entropy loss is widely used as the loss function for training MLDC models. Suppose y is the ground truth label and p is the predicted probability, then the binary cross entropy loss is $L_{BCE}(p_t) = -\log p_t$.

To tackle the long-tail distribution of the labels, we also apply the focal loss (Lin et al., 2017), which adds a weight term to the ordinary binary cross entropy loss to dynamically down-weights the loss assigned to well-classified labels. The focal loss is

$$L_{FL}(p_t) = -(1 - p_t)^\gamma \log p_t \tag{7}$$

Here γ is a tunable parameter to adjust the strength of down-weighting. The weight term $(1-p_t)^\gamma$ suppresses the loss from well-classified labels (where p_t is high) and bias the model towards labels that get wrong predictions.

In practice, using the focal loss from the beginning of training isn't ideal, because it tends to correct the misclassified rare labels while sacrificing the performance on the frequent labels. Instead, we first train our model with the ordinary binary cross entropy loss to allow the model to learn general features and perform well on frequency labels. Once the model performance saturates, we switch

Dataset	N_{train}	N_{val}	N_{test}	$\overline{N_w}$	N_l	$\overline{N_l}$
MIMIC-III-full	47,724	1,632	3,372	1,485	8,922	15.9
MIMIC-III-50	8,067	1,574	1,730	1,530	50	5.7
Dutch	2,511	313	313	4,958	144	5
French	22,540	2,836	2,827	1,660	940	11

Table 1: Data statistics. N_{train}: number of training instances, N_{val}: number of validation instances, N_{test}: number of test instances, $\overline{N_w}$: average number of words per instance, N_l: number of labels in total, $\overline{N_l}$: average number of labels per instance.

to use the focal loss and further fine-tune the model to improve the predictions on rare labels.

4 Experiments

4.1 Datasets

We evaluated our model on the widely used medical benchmark dataset MIMIC-III, as well as two medical datasets in Dutch and French respectively. The statistics of the datasets are listed in Table 1.

4.1.1 MIMIC-III

The **M**edical **I**nformation **M**art for **I**ntensive **C**are III (MIMIC-III) is an open-access dataset comprised of hospital records associated with over 4000 patients. We focus on using the discharge summaries to predict their tagged International Classification of Diseases 9 (ICD-9) codes. We formulate this task as a MLDC problem following prior work (Shi et al., 2017; Mullenbach et al., 2018). In total, there are 52,722 discharge summaries and 8,922 unique ICD-9 codes. We follow the experiment settings of Mullenbach et al. (2018). We focus on the experiment that predicts the full 8,922 ICD-9 codes (denoted as MIMIC-III-full) but also present the results on the top-50 ICD-9 codes (denoted as MIMIC-III-50). The data statistics of the two experiments are listed in Table 1.

4.1.2 Dutch and French Datasets

Many European hospitals are aware of the advantages of automatic coding solutions that improve the accuracy and efficiency of medical coding. To evaluate how well our model adapts to coding on non-English medical documents, we use two real-world datasets, one in Dutch and the other in French. These datasets contain human assigned ICD-10 codes for each encounter. In these datasets, the Discharge Summary is not differentiated from other documents so we concatenate all documents in the encounter. Nonetheless, the French data has similar encounter length to MIMIC-III, but the Dutch data are much longer with an average of 30 documents or close to 5,000 tokens per encounter.

The ICD-10 code system is widely used in European countries, but no benchmark dataset is available for comparing coding methods – likely due to existing patient data protection regulations in the EU. For U.S. English data, the restrictions are somewhat less, which is how MIMIC-III was able to be produced – though still at immense de-identification cost. Although the de-identification and release of the French/Dutch data was not possible, we believe our experiments and findings still benefit the research community because they (1) demonstrate that our model can generalize to other languages and (2) are the first medical coding results reported for French or Dutch.

4.2 Preprocessing and Hyperparameters

We follow the preprocessing schema of (Mullenbach et al., 2018) except that we keep numerical values from one to ten as they are relevant for coding. We utilize the word2vec CBOW method to pretrain the word embeddings of size $d_e = 100$ and 200 on the preprocessed texts for MIMIC-III and the non-English sets respectively. All MIMIC documents are truncated to a maximum sequence length w_{max}=3,500, whereas both 2,500 and 3,500 sequence length were used for the non-English sets.

We found optimal hyperparameter settings using the Ray Tune library (Liaw et al., 2018). We optimized values for out-channel size d_{conv} and filter size k of the convolutional layer in each SE module, dropout probability q after the input embedding layer, as well as the power term γ in the focal loss function. To reduce the search space, we set $d^1_{conv} = d^2_{conv}$, $d^3_{conv} = d^4_{conv}$ and $k^1 = k^2$, $k^3 = k^4$. Table 2 summarizes their optimal values for different experiments. We use four Res-SE blocks across all experiments, and adopt the Adam optimizer with an initial learning rate of 0.00015.

4.3 Evaluation Metrics

The goal of computer assisted coding is to have as little human intervention as possible. This means that a model trained for coding should aim to predict the correct codes from the full set rather than the top N codes, or give a ranked list of possible codes. The performance of a model on the top 50 codes is often reported in research papers. However, in real-world settings, top-50 metrics are insufficient for making an accurate assessment of automatic coding because expensive human resources

	d^i_{conv}	k^i	q	γ
Range	100-240	5-25	0.1-0.3	0.5-2
MIMIC-III-full	200,200, 240,240	13,13, 9,9	0.3	1
MIMIC-III-50	180,180, 200,200	11,11, 9,9	0.3	0.5
Dutch and French	180,180, 200,200	11,11, 9,9	0.3	0.5

Table 2: The parameter values used in different tasks. d^i_{conv}, k^i: the out-channel size and the kernel size of the SE convolutional layer in the i^{th} Res-SE block, q: the dropout probability after the input embedding layer, γ: the power term in the focal loss.

are still needed for the large number of remaining codes. In MIMIC-III, top 50 codes cover only a third of the codes per encounter, and in reality a small number of top codes can usually be handled by rule-based systems with great accuracy.

Ranking based metrics like P@K, R@K, RP@K (Chalkidis et al., 2020), where K is often the average number of labels per document, are rarely used in coding because there is high variability in the number of codes per encounter. In MIMIC-III, the number of codes per encounter varies from one to 79, and 43% of the encounters have more than the average 15 codes. Asking a human coder to always review K codes for every encounter would cause a huge productivity drop because she will still have to review K codes when there is only one code. On the other hand, reducing the number of gold codes to K (Chalkidis et al., 2019) will result in inaccurate measures (especially for Recall) for a large percentage of encounters with more than K codes and artificially inflate system performance.

Although macro metrics are useful for assessing performance on rare codes, they are less important in determining overall coding performance. For these reasons, micro precision, recall and F1 over all codes best reflect improvements in coding productivity because they directly measure the accuracy and coverage of the code assignment by models. However, prior work did not report precision and recall on the MIMIC data. For comparison purposes, we report F1 and other previously used metrics on both MIMIC-III and the non-English datasets, but the emphasis should be on Micro F1.

5 Results

To evaluate the effectiveness of our methods, we compare our model with the existing state-of-the-art. The results shown below are generated from the average of five runs with different random seeds for parameter initialization. We also investigate the interpretability of the model.

5.1 Results on MIMIC-III

Table 3 shows the results on the MIMIC-III dataset using the full ICD-9 codes. Our model achieved the strongest results across multiple metrics compared to the other systems. In particular, our model improves the state-of-the-art Micro F1 score as well as ranking based precision scores. Table 3 also shows that the systems achieved very similar results on Micro AUC for all codes even when they differ significantly in other metrics. This suggests that Micro AUC is not sensitive enough to distinguish different systems and is therefore not a good metric for comparing coding models.

Table 4 shows the results for the top-50-code prediction. Our model produced competitive results with other top models.

An interesting observation is that multi-layer attention yields better results on MIMIC-III-50 but sum-pooling attention performs better on MIMIC-III-full. One possible explanation is that when there are sufficient training data for the labels, multi-layer attention with more parameters is able to learn better representations for each label. Whereas when the data is insufficient given the label size, aggregating information over labels yields better results.

5.2 Results on Dutch and French

On the Dutch and French datasets, we establish two baselines. The first is MultiResCNN (Li and Yu, 2020), which is the best performing model on MIMIC-III that is publicly available. The second is XLM-RoBERTa (Conneau et al., 2019), a multi-lingual transformer model.[1] XLM-RoBERTa and related models achieve excellent performance on well-known benchmarks such as GLUE (Wang et al., 2018), however they are not well established on the task of long-document, multi-label classification. Table 5 presents our results.

Of the models we considered, only Effective-CAN can be trained on the full label set (i.e. 144 codes for Dutch, 940 codes for French): XLM-RoBERTa and MultiResCNN run out of 16GB GPU memory. As such, we resort to comparison with the baselines on only the top-50 codes. XLM-RoBERTa yields poor results for both Dutch and

[1] We use the implementation available from HuggingFace (Wolf et al., 2020).

Model	AUC		F1		P@k	
	Macro	Micro	Macro	Micro	8	15
CAML (Mullenbach et al., 2018)	0.895	0.986	0.088	0.539	0.709	0.561
DR-CAML (Mullenbach et al., 2018)	0.897	0.985	0.086	0.529	0.690	0.548
MSATT-KG (Xie et al., 2019)	0.910	**0.992**	0.090	0.553	0.728	0.581
MultiResCNN (Li and Yu, 2020)	0.910	0.986	0.085	0.552	0.734	0.584
HyperCore (Cao et al., 2020)	**0.930**	0.989	0.090	0.551	0.722	0.579
LAAT (Vu et al., 2020)	0.919	0.988	0.099	0.575	0.738	0.591
JointLAAT (Vu et al., 2020)	0.921	0.988	**0.107**	0.575	0.735	0.590
EffectiveCAN (Multi-layer attention)	0.921	0.989	0.105	0.581	0.755	0.604
EffectiveCAN (Sum-pooling attention)	0.915	0.988	0.106	**0.589**	**0.758**	**0.606**

Table 3: Results on MIMIC-III-full (i.e. all codes)

Model	AUC		F1		P@k
	Macro	Micro	Macro	Micro	5
C-LSTM-Att (Shi et al., 2017)	-	0.900	-	0.532	-
CAML (Mullenbach et al., 2018)	0.875	0.909	0.532	0.614	0.609
DR-CAML (Mullenbach et al., 2018)	0.884	0.916	0.576	0.633	0.618
MSATT-KG (Xie et al., 2019)	0.914	0.936	0.638	0.684	0.644
MultiResCNN (Li and Yu, 2020)	0.899	0.928	0.606	0.670	0.641
HyperCore (Cao et al., 2020)	0.895	0.929	0.609	0.663	0.632
LAAT (Vu et al., 2020)	**0.925**	**0.946**	0.666	0.715	**0.675**
JointLAAT (Vu et al., 2020)	**0.925**	**0.946**	0.661	0.716	0.671
EffectiveCAN (Multi-layer attention)	0.920	0.945	**0.668**	**0.717**	0.664
EffectiveCAN (Sum-pooling attention)	0.915	0.938	0.644	0.702	0.656

Table 4: Results on MIMIC-III-50 (i.e. top-50 codes only)

French. Recall is particularly low, likely caused by the model only seeing the first 512 subwords of a long encounter with thousands of tokens.

Our model with multi-layer attention substantially outperforms the other two systems. It strikes a good balance between precision and recall, and is able to handle the full code sets without difficulties. Unlike the observation of Li and Yu (2020) where the maximum length didn't make an obvious difference to the performance on MIMIC-III, we found that training on longer sequences on Dutch and French gives an extra boost to all metrics. This is especially true for the Dutch which contains longer encounter texts. The results show that EffectiveCAN can be easily retrained for non-English documents to very good effect.

5.3 Analysis of Focal Loss

In this section, we describe our experiments on the MIMIC-III-full dataset for a better understanding of the focal loss.

To investigate how the moment of loss function switch impacts model performance, we trained models with focal loss activated at different training epoch and the results are given in Table 6. It shows that switching the loss function at a later stage yields more pronounced improvement in Macro F1. We obtained the best results by training with BCE loss first and saving the best model as measured by the micro-F1 on the dev set. Then we continued training using the focal loss until it converged.

To better understand which tail labels the focal loss helps improve, we analyzed model performance based on label frequency in the test set. Table 7 shows that the focal loss improves the prediction of both frequent and rare labels, but the improvement is more pronounced for the less frequent labels.

5.4 Discussion

In this section we analyze the differences between the models. Compared to CAML, MultiResCNN yields better results by enhancing the encoder using the multi-filter residual convolutional network, and HyperCore improves the macro-metrics by incorporating the correlations within the labels. Although both MSATT-KG and EffectiveCAN use multi-layer attention, we differ in the ways of aggregating the attention results. Our model uses all the attended values for the final label prediction whereas MSATT-KG performs extra max-pooling operations before the prediction. The max-pooling operations, in our opinion, are unnecessary and risk losing information. Our model produces notably better results than MSATT-KG on the full code set.

JointLAAT differs from EffectiveCAN in the encoder layer where it uses the BiLSTM to capture contextual information, whereas we choose to

Model	Dutch				French			
	# Labels	Precision	Recall	F1	# Labels	Precision	Recall	F1
XLM-RoBERTa	50	0.725	0.289	0.413	50	0.606	0.426	0.500
MultiResCNN	50	0.458	0.639	0.534	50	0.631	0.607	0.619
EffectiveCAN	50	0.822	0.760	0.790	50	0.692	0.620	0.654
EffectiveCAN (3,500)	50	**0.873**	**0.777**	**0.822**	50	**0.705**	**0.636**	**0.669**
EffectiveCAN (3,500)	144	**0.844**	**0.732**	**0.784**	940	**0.583**	**0.493**	**0.534**

Table 5: Results on Dutch and French

Training	F1	
Epoch	Macro	Micro
0	0.084	0.578
4	0.094	0.581
8	0.099	0.587
11	0.106	0.588

Table 6: Moment of the loss function swith

Label	F1 w/o Focal Loss		F1 w Focal Loss	
Frequency	Macro	Micro	Macro	Micro
0-10	0.139	0.301	0.155	0.322
11-50	0.407	0.490	0.426	0.514
51-100	0.521	0.568	0.528	0.578
101-200	0.578	0.626	0.606	0.646
over 200	0.698	0.751	0.699	0.753

Table 7: Effect of focal loss by label frequency in the test set

use the convolution-based model for computational and memory efficiency. To deal with rare labels, prior works often add a separate component such as a graph neural network or a hierarchical joint learning module, which inevitably increases the complexity and size of the model. Instead, we employ the focal loss, which can be easily modified from the binary cross entropy loss, to improve the rare-label prediction without sacrificing the overall performance. By refining the entire model structure including the convolutional encoder, attention coverage and training objective, we build a model that is simple and easy to scale, yet very effective for the medical coding problem. The model achieved the best micro F1 results on the MIMIC-III dataset, even when compared with more complex models. It is capable of not only generating accurate top codes but also covering a large number of codes including rare codes, which is important for real world applications in the medical domain.

Recent results (You et al., 2019b; Chalkidis et al., 2020) show that RNN-based and BERT-based models performed well on the topic categorization tasks of EUR-LEX, AMAZON, WIKIPEDIA and RCV1. However, it's also clear that the best models on these tasks are typically *not the same* as the best performing models on MIMIC-III, which is fundamentally not a topic categorization task. Rather medical coding requires fine-grained analysis of very narrow aspects of the document in order to identify appropriate codes. For an additional point of comparison, we evaluated EffectiveCAN on two topic categorization tasks (EUR-Lex and Wiki10-31K) and found it outperforms several strong baselines and is only lower than X-Transformer (Chang et al., 2020), a large pre-trained transformer model, by a small margin on most metrics. Detailed results are reported in Appendix A.

5.5 Model Interpretability

It is a requirement of medical coding that an automatic coding system is able to extract text evidence to support the generated billing codes. With the attention mechanism, we can extract the text snippets that support the predicted codes. More specifically, by conducting the multi-layer attention on the four Res-SE layer outputs, we obtain four attention weight matrices $A^{i\in\{1,2,3,4\}}$ with each $A^i \in \mathbb{R}^{N_l \times N_w}$. For the j^{th} label, the associated attention weights are the j^{th} column of each matrix, that is $A^i_{\cdot j} \in \mathbb{R}^{N_w}$. Next, to get the most influential text span for the j^{th} label, we first get the text position k^* which is the argmax of all attention weights:

$$k^* = \underset{k}{\operatorname{argmax}}\{A^1_{kj}, A^2_{kj}, A^3_{kj}, A^4_{kj}\} \qquad (8)$$

We then select the most informative n-gram features surrounding the text position k^*.

Table 8 gives some examples of the extracted text snippets for the predicted ICD-9 codes in the MIMIC-III-full experiments. Our model is able to extract the n-gram features that are similar to the code descriptions, e.g., the extracted snippet *"Systolic congestive heart failure"* for 428.20. More importantly, our model is capable of selecting phrases with different syntactic forms but similar semantics as the code descriptions, e.g., the extracted snippet *"percutaneous tracheostomy tube placement"* for 934.1. It indicates that the model can learn inter-

ICD-9 code & Description	Document texts
934.1: “Foreign body in main bronchus”	. . . During his ICU stay he underwent **percutaneous tracheostomy tube placement** as well as . . .
428.20: “Systolic heart failure, unspecified”	. . . Primary Diagnosis:1. Anterior ST elevation myocardial infarction. 2. **Systolic congestive heart failure**. 3. Atrial fibrillation. . .
784.2: “Swelling, mass, or lump in head and neck”	. . . dexamethasone once daily, to **reduce brain swelling after** the bleeding and keppra twice daily. . .
585.9: “Chronic kidney disease, unspecified”	. . . Patient likely had acute on chronic renal with **chronic renal dysfunction** secondary to. . .

Table 8: Examples of model interpretability. The extracted n-gram features are highlighted in bold face.

Model	F1	
	Macro	Micro
EffectiveCAN w Multi-layer attention	0.105	0.581
w/o residual module	0.097	0.573
w/o SE module	0.101	0.576
only attend to the first Res-SE layer	0.093	0.572
only attend to the last Res-SE layer	0.086	0.567
w/o focal loss	0.095	0.577

Table 9: Ablation study of the multi-layer attention model

Model	F1	
	Macro	Micro
EffectiveCAN w Sum-pooling attention	0.106	0.589
w/o residual module	0.102	0.580
w/o SE module	0.076	0.506
w/o focal loss	0.101	0.575

Table 10: Ablation study of the sum-pooling attention model

pretable representations from the input and capture the informative evidence for each code.

6 Ablation Study

We conducted ablation studies to verify the effectiveness of each module in our model. We compare the results on MIMIC-III-full between the ordinary model and the one with a component removed. The results for the macro- and micro-F1 scores are listed in Table 9.

For the multi-layer attention model, removing the residual module causes a notable reduction in both the macro- and micro-F1 scores, indicating the importance of the residual module in the deep convolutional encoder of our model. Meanwhile, the model without the SE module also reports a lower macro-F1 and micro-F1, which implies that the SE module enables the model to produce better representations for the predictions.

Only attending to the first or last Res-SE layer output leads to worse results. It confirms our argument that the multi-layer attention can capture information from the input at different levels, which further facilitates better predictions. It is also possible to completely remove the attention module, but since (Mullenbach et al., 2018) has shown that label-wise attention improves F1, this experiment wasn’t deemed informative.

Compared to the original model, the one without using the focal loss produces a slightly lower result in the micro-F1 but a large reduction in the macro-F1. This verifies the effectiveness of the focal loss in tackling the long-tail distribution of the labels.

For the sum-pooling attention model, removing the SE module results in the largest performance drop. We have yet to find an explanation for this difference in the two attention models.

7 Conclusions

In this paper, we proposed an effective convolutional attention network for MLDC, and showed its effectiveness for medical coding on long documents. Our model features a deep and more refined convolutional encoder, consisting of multiple Res-SE blocks, to capture the multi-scale patterns of the document texts. Furthermore, we use the multi-layer attention to adaptively select the most relevant features for each label. We employ the focal loss to improve the rare-label prediction without sacrificing the overall performance. Our model obtains the state-of-the-art results across several metrics on MIMIC-III, and compares favorably with other systems on two non-English datasets.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Rohit Babbar and Bernhard Schölkopf. 2017. Dismec: Distributed sparse machines for extreme multi-label classification. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 721–729.

Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Shengping Liu, and Weifeng Chong. 2020. Hypercore: Hyperbolic and co-graph representation for automatic icd coding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3105–3114.

Ilias Chalkidis, Emmanouil Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2019. Extreme multi-label legal text classification: A case study in EU legislation. In *Proceedings of the Natural Legal Language Processing Workshop 2019*, pages 78–87, Minneapolis, Minnesota. Association for Computational Linguistics.

Ilias Chalkidis, Manos Fergadiotis, Sotiris Kotitsas, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. An empirical study on large-scale multi-label text classification including few and zero-shot labels. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7503–7515, Online. Association for Computational Linguistics.

Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S Dhillon. 2020. Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3163–3171.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Lu Hou, Jinhua Zhu, James Kwok, Fei Gao, Tao Qin, and Tie-yan Liu. 2019. Normalization helps training of quantized lstm. In *Advances in Neural Information Processing Systems*, pages 7346–7356.

Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9.

Fei Li and Hong Yu. 2020. Icd coding from clinical text using multi-filter residual convolutional neural network. In *AAAI*, pages 8180–8187.

Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E Gonzalez, and Ion Stoica. 2018. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*.

Junyang Lin, Qi Su, Pengcheng Yang, Shuming Ma, and Xu Sun. 2018. Semantic-unit-based dilated convolution for multi-label text classification. *arXiv preprint arXiv:1808.08561*.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jueqing Lu, Lan Du, Ming Liu, and Joanna Dipnall. 2020. Multi-label few/zero-shot learning with knowledge aggregated from multiple label graphs. *arXiv preprint arXiv:2010.07459*.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*.

Ankit Pal, Muru Selvakumar, and Malaikannan Sankarasubbu. 2020. Multi-label text classification using attention-based graph neural network. *arXiv preprint arXiv:2003.11644*.

Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. 2018. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *Proceedings of the 2018 World Wide Web Conference*, pages 993–1002.

Haoran Shi, Pengtao Xie, Zhiting Hu, Ming Zhang, and Eric P Xing. 2017. Towards automated icd coding using deep learning. *arXiv preprint arXiv:1711.04075*.

Yukihiro Tagami. 2017. Annexml: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 455–464.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2020. A label attention model for icd coding from clinical text. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xiancheng Xie, Yun Xiong, Philip Yu, and Yamgyong Zhu. 2019. 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*.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. Sgm: sequence generation model for multi-label classification. *arXiv preprint arXiv:1806.04822*.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019a. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. In *Advances in Neural Information Processing Systems*, pages 5820–5830.

Yongjian You, Weijia Jia, Tianyi Liu, and Wenmian Yang. 2019b. Improving abstractive document summarization with salient information modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2132–2141, Florence, Italy. Association for Computational Linguistics.

A Appendix

A.1 Additional Experiments

We also evaluated our model on two large-scale benchmark datasets: EUR-Lex and Wiki10-31K, to show the effectiveness of our model across domains. We use $w_{max} = 2000, 3000$ for EUR-Lex and Wiki10-31K respectively, and the hyperparameters of the models are given in Table 11.

A.1.1 Datasets

EUR-Lex consists of a collection of documents of European Union laws. It contains 19,314 documents in total with 3,956 categories regarding different aspects of European law. We follow the setting of You et al. (2019a) to split the train and test sets, obtaining 15,449 and 3,865 training and testing documents. From the training set, we then take 1,545 documents out for validation, resulting in 13,904 training documents.

Wiki10-31K is a collection of social tags for Wikipedia pages. It's composed of 20,762 documents and 30,938 associated tags. We also use the setting of You et al. (2019a) to get 14,146 and 6,616 training and testing documents. We then use 1415 documents for validation, resulting in 12,731 training documents.

A.1.2 Results

The results on the EUR-Lex dataset are listed in Table 13. The results from our model are higher than some strong baselines including AnnexML (Tagami, 2017), DiSMEC (Babbar and Schölkopf, 2017), Parabel (Prabhu et al., 2018), and AttentionXML (You et al., 2019a), and is only lower than X-transformer (Chang et al., 2020) by a tiny gap, e.g. 0.08% lower on precision@1. We also observe that traditional ML models, such as AnnexML, DiSMEC and Parabel, generally produce worse results than deep learning model such as AttentionXML. By employing large-scale pretrained transformer-based models, X-tranformer reports the start-of-the-art results.

Table 13 also shows that our model produces very competitive results on the Wiki10-30K dataset. Our model outperforms most baselines except for X-transformer. The losing margins are quite small, 0.66% on precision@1, 0.08% on precision@3, and 0.43% on precision@5.

Compared to the large-scale transformer-based models, our model is more effective in terms of balancing the model performance and model size. Table 12 lists the comparison of the model size between our model, AttenionXML, and the transformer-based models used in X-transformer. We can see that our model is much smaller than BERT-large, XLNet-large and Roberta-large used in X-transformer. Note that there are other components in X-transformer that we don't take into account. With a significantly smaller model size, our model achieved less than 1% drop on EUR-Lex and Wiki10-31K datasets compared to X-transformer. In addition, our model can handle much longer sequences than transformer models (maximum 512 tokens). This is especially important when the information for predicting labels is spread over the long document.

	d^i_{conv}	k^i	q	γ
Range	100-240	5-25	0.1-0.3	0.5-2
EUR-Lex	180,180, 200,200	15,15, 7,7	0.1	1
Wiki10-31K	160,160, 220,220	17,17, 5,5	0.3	1

Table 11: The parameter values used in different tasks. d^i_{conv}, k^i: the out-channel size and the kernel size of the SE convolutional layer in the i^{th} Res-SE block, q: the dropout probability after the input embedding layer, γ: the power term in the focal loss.

Model	# parameters	Model size
AttentionXML, EUR-Lex	-	0.20GB
AttentionXML, Wiki10-31K	-	0.62GB
BERT-large	340M	-
RoBERTa-large	355M	-
XLNet-large	340M	-
EffectiveCAN, EUR-Lex	10M	0.12GB
EffectiveCAN, Wiki10-31K	38M	0.46GB

Table 12: Model size comparison between EffectiveCAN, AttentionXML and the transformer-based models (BERT-large, RoBERTa-large, XLNET-large) used in X-transformer

Model	EUR-Lex			Wiki10-31K		
	P@1	P@3	P@5	P@1	P@3	P@5
AnnexML (Tagami, 2017)	79.66	64.94	53.52	86.46	74.28	64.20
DiSMEC (Babbar and Schölkopf, 2017)	83.21	70.39	58.73	84.13	74.72	65.94
Parabel (Prabhu et al., 2018)	82.12	68.91	57.89	84.19	72.46	63.37
AttentionXML (You et al., 2019a)	87.12	73.99	61.92	87.47	78.48	69.37
X-Transformer (Chang et al., 2020)	**87.22**	**75.12**	**62.90**	**88.51**	**78.71**	**69.62**
Our EffectiveCAN	87.14	74.28	61.95	87.85	78.63	69.29

Table 13: Results on EUR-Lex and Wiki10-31K (values in percentage)