Don't Just Pay Attention, PLANT It: Transfer L2R Models to Fine-tune Attention in Extreme Multi-Label Text Classification For ICD Coding

Anonymous ACL submission

Abstract

The keystone of state-of-the-art Extreme Multi-Label Text Classification (XMTC) models is the multi-label attention layer within the decoder, which deftly directs label-specific focus to salient tokens in input text. Nonetheless, the process of acquiring these optimal attention weights is onerous and resource-intensive. To alleviate this strain, we introduce PLANT – **P**retrained and **L**everaged **A**tte**NT**ion – an innovative transfer learning strategy to fine-tune XMTC decoders. The central notion involves transferring a pretrained learning-to-rank (L2R) model, utilizing its activations as attention weights, thereby serving as the *'planted'* attention layer in the decoder. On the full MIMIC-III dataset, PLANT excels in four out of seven metrics and surpasses in five for the top-50 code set, demonstrating its effectiveness. Remarkably, for the rare-50 code set, PLANT achieves a significant $12.7-52.2\%$ improvement in four metrics. On MIMIC-IV, it leads in three metrics. Notably, in low-shot scenarios, PLANT matches traditional attention models' precision despite using significantly less data ($\frac{1}{10}$ for precision at 5, $\frac{1}{5}$ for precision at 15), highlighting its efficiency with skewed label distributions.

1 Introduction

Extreme Multi-Label Text Classification (XMTC) addresses the problem of automatically assigning each data point with most relevant subset of labels from an extremely large label set, often containing hundreds of thousands, even millions of labels and samples in various real-world XMTC applications. One major application of XMTC is in the global healthcare system, specifically in the context of the International Classification of Diseases (ICD)[1]. ICD coding is the process of assigning codes representing diagnoses and procedures performed during a patient visit using clinical notes documented by health professionals (Table 1). ICD codes are used for both epidemiological studies and billing of services (Bottle and Aylin, 2008). XMTC has been utilized to automate the manual ICD coding performed by clinical coders which is time intensive and prone to human errors (O'malley et al., 2005; Nguyen et al., 2018).

998.32 : *Disruption of external operation wound*
··· wound infection, and **wound breakdown** ···
428.0 : *Congestive heart failure*
··· DIAGNOSES: 1. **Acute congestive heart failure** 2. Diabetes mellitus 3. Pulmonary edema ···
202.8 : *Other malignant lymphomas*
··· a 55 year-old female with **non Hodgkin's lymphoma** and acquired C1 esterase inhibitor deficiency ···
770.6 : Transitory tachypnea of newborn
··· Chest x-ray was consistent with **transient tachypnea of the newborn** ···
424.1 : *Aortic valve disorders*
··· mild **aortic stenosis with an aortic valve area** of 1.9 cm squared and 2+ **aortic insuffiency** ···

Table 1: Examples of clinical text fragments and their corresponding ICD codes (Li and Yu, 2020).

Main Challenge: Building XMTC models is challenging because datasets often consist of texts with multiple lengthy narratives – more than 1500 tokens (i.e., words) on average. However, only a small fraction of tokens are most informative with regard to assigning relevant labels. Automatically assigning labels become even more challenging when, (1) the label space is extremely high dimensional, and, (2) the label distribution is heavily skewed. For example, in automatic ICD coding, there are over 18000 and 170000 codes in ICD-9-CM and ICD-10-CM/PCS[2], respectively. The skewness of ICD-9-CM label distribution in the MIMIC-III dataset (Johnson et al., 2016) is evident from the fact that approximately 5411 out of all the 8929 codes appear less than 10 times (refer to Appendix A.1, Figure 5 for a visual).

[1] https://www.who.int/standards/classifications/classification-of-diseases

[2] https://www.cdc.gov/nchs/icd/icd10cm_pcs_background.htm

How SOTA models address the main challenge in XMTC? (Red Box) In state-of-the-art (SOTA) NLP models, the inclusion of *attention* mechanisms is crucial, benefiting various applications like Machine Translation, Summarization, Text Representation, Sentiment Analysis, and Question Answering (Vaswani et al., 2017; Tang et al., 2018; Xu et al., 2020; Kiela et al., 2018; Wang et al., 2020; Dehghani et al., 2018). In XMTC, these attention mechanisms play a vital role in addressing the challenges of high-dimensional label spaces and skewed label distributions. XMTC models (Mullenbach et al., 2018; Xie et al., 2019; Li and Yu, 2020; Cao et al., 2020; Vu et al., 2021; Zhou et al., 2021; Liu et al., 2021; Yuan et al., 2022; Zhang et al., 2022; Yang et al., 2022) consistently feature a multi-label attention layer, dynamically allocating label-specific attention weights to the most informative tokens in input text. Refer to the components highlighted in red in Figure 1, which illustrate this critical attention layer in action. Regardless of the specific encoder architecture, removing this attention layer leads to a significant drop in performance.

Main Shortfall in Red Box: Current SOTA XMTC models often begin with random attention weights, requiring them to rank all tokens for each label from scratch, a computationally intensive process, which, given the high-dimensional label space characteristic of XMTC datasets, leads to extensive computational requirements and prolonged training times. Moreover, the presence of heavily skewed label distributions further exacerbates this challenge, as rare labels necessitate even longer training durations and increase the risk of overfitting (Figure 4). Corroborating the issue of rare codes, the study in (Edin et al., 2023) reveals that SOTA models exhibit considerable difficulties when predicting rare ICD diagnosis codes (Figure 2). Models tend to perform similarly across codes with comparable frequencies, implicating the higher proportion of rare codes in ICD as a significant factor in performance disparities. Correlations between code frequency and F1 score are moderately high, indicating that rare codes are predicted with less accuracy than common ones. This inherent complexity underscores the need for efficient mechanisms to learn optimal attention configurations in XMTC models, as starting with random weights may not suffice.

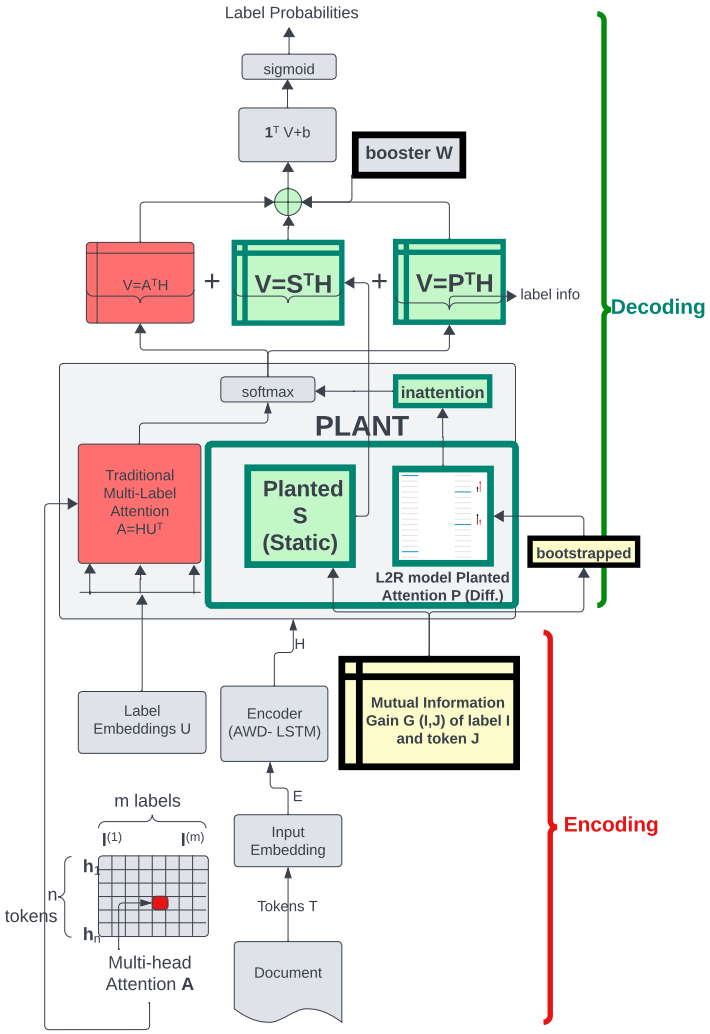

Figure 1: Architecture of PLANT showcasing the integration of contemporary SOTA components (grey box), multi-label attention (red box), planted attention (green box), and mutual information gain (yellow box) to enhance label prediction efficacy.

Our Contributions (Green Box):

1. To address this shortfall, we propose PLANT – **P**retrained and **L**everaged **A**tte**NT**ion, a novel transfer learning mechanism to fine-tune attention in XMTC. The core idea is to bootstrap using mutual information gain (refer to the yellow components in Figure 1) a standalone model that learns to rank (L2R) tokens based on their relevance to labels. The pretrained L2R model (refer to the green components in Figure 1) that leverages its activations as attention weights serves as the *'planted'* attention layer in the XMTC decoder. This tranferring of the L2R model ensures the decoder starts

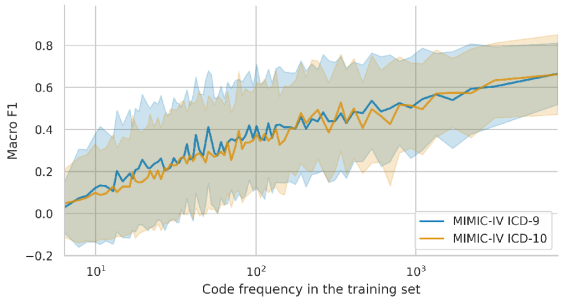

Figure 2: Comparative analysis of model performance from (Edin et al., 2023) on rare versus common ICD diagnosis codes, highlighting that rare codes have near zero macro-F1scores.

with well-informed attention weights rather than training from scratch with randomly initialized weights. Subsequent fine-tuning enables not only efficient convergence toward optimal attention weight configurations but also enhances the model's ability to prioritize salient features of the input texts while minimizing the risk of overfitting by adapting them to the specific characteristics of the target dataset. Notably, we compared PLANT with a SOTA model LAAT (Vu et al., 2021) on `MIMIC-IV-full`, showing that PLANT avoids overfitting during training (Figure 4).

2. Addressing the shortfall in rare code prediction (Edin et al., 2023), PLANT is particularly effective in dealing with high dimenstional skewed label distributions in a low shot setting. It demonstrates comparable precision to traditional attention models, even with substantially less data – $\frac{1}{10}$for precision at 5, $\frac{1}{5}$ for precision at 15 (Figure 3).
3. We introduce the *inattention* technique, which strategically filters out less relevant tokens, enhancing the significance of attention weights and enabling a sharper focus on critical elements within a token sequence. Additionally, inspired by Backpropagation-Through-Time-for-Text-Classification (Howard and Ruder, 2018), we propose a *stateful decoder* that accumulates information across segments, enabling cumulative predictions. This mechanism utilizes batch-level states, improving adaptability to large documents and model convergence, eliminating text truncation needs, and ensuring stable GPU memory usage, thereby enhancing both performance and efficiency (Table 7).
4. We extensively evaluated PLANT on benchmark MIMIC-III and newly available MIMIC-IV datasets, widely used in automatic ICD coding research. Compared to 10 existing SOTA models (Section 3.2), PLANT outperformed them across 7 different evaluation metrics. Specifically, in `MIMIC-III-full`, `MIMIC-III-top50`, `MIMIC-III-rare50`, and `MIMIC-IV-full` datasets, PLANT exhibited significant performance improvements (Table 3, Table 4, Table 5, Table 6). We also conducted rigorous ablation analysis (Section 4) and made our trained models and code available at `https://anonymous.4open.science/r/brainsplant/`.

2 Approach

XMTC: The input is a set of documents and their corresponding labels, $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid \boldsymbol{y}_i \in \{0,1\}^{|\mathcal{L}|}, i = 1, \ldots, d\}$, where $\mathcal{L}$ is the set of labels. The goal of XMTC is to learn a prediction function $\hat{y}(\boldsymbol{x}_i) \in \mathbb{R}^{|\mathcal{L}|}$. The function $\hat{y}$ should be optimized such that the $\hat{y}(x_{il})$ is high when $y_{il} = 1$ (i.e., label l is relevant to $\boldsymbol{x}_i$), and is low when $y_{il} = 0$.

Intuition behind our XMTC model - PLANT (Figure 1): The intuitive flow starts with document tokenization into embeddings processed by a pretrained AWD-LSTM to grasp textual contexts. The decoder introduces *planted attention* (green box), leveraging a L2R model's ability to rank token significance by label relevance, enriching the model with a pre-understanding of token-label dynamics. This is adeptly paired with multi-label attention (red box), merging learned and pretrained insights for feature prominence. Additionally, mutual information gain (yellow box) is utilized to enhance the decision-making process by calculating the relevance of each token to the potential labels, providing an informed basis for further attention refinement. A subsequent boost attention phase fine-tunes this for label-specific discernment, culminating in a sigmoid-derived label probability prediction. Section 2.1 provides a detailed description of the L2R model components, while Section 2.2 explains how we utilize the pretrained L2R model for planted attention, illustrating the integration of the green boxes in Figure 1.

2.1 L2R Model

We use superscript to denote the id of a label and subscript to denote the id of a token. The training set of the L2R model contains a set of labels $\mathcal{L} = \left\{l^{(1)}, l^{(2)}, \cdots, l^{(m)}\right\}$, and a set of tokens $\mathcal{T} = \{t_1, t_2, \cdots, t_n\}$. Furthermore, $\boldsymbol{G} = \left[\boldsymbol{g}^{(1)}, \boldsymbol{g}^{(2)}, \cdots, \boldsymbol{g}^{(m)}\right] \in \mathbb{R}^{n \times m}$, and $\boldsymbol{g^{(i)}} = \left[g_1^{(i)}, g_2^{(i)}, \cdots, g_n^{(i)}\right]^T \in \mathbb{R}^n$, where $g_j^{(i)}$ denotes the relevance of the token t_j with respect to label $l^{(i)}$. We represent each label $l^{(i)}$ and token t_j with word embeddings $\boldsymbol{e}_{l^{(i)}}$ and $\boldsymbol{e}_{t_j}$, respectively. A feature vector

$$\boldsymbol{x_j^{(i)}} = \Psi\left(\boldsymbol{e}_{l^{(i)}}, \boldsymbol{e}_{t_j}\right) \tag{1}$$

is created from each label-token pair $\left(l^{(i)}, t_j\right), i = 1, 2, \cdots, m; j = 1, 2, \cdots, n$, by concatenating the corresponding word embeddings $\boldsymbol{e}_{l^{(i)}}$ and $\boldsymbol{e}_{t_j}$. The feature matrix, $\boldsymbol{X^{(i)}} = \left[\boldsymbol{x_1^{(i)}}, \cdots, \boldsymbol{x_n^{(i)}}\right]$ and the corresponding scores, $\boldsymbol{g^{(i)}} = \left[g_1^{(i)}, g_2^{(i)}, \cdots, g_n^{(i)}\right]^T$ then form an 'instance'. The training set can be denoted as $\left\{\left(\boldsymbol{X^{(i)}}, \boldsymbol{g^{(i)}}\right)\right\}_{i=1}^{m}$. The L2R model is associated with a ranking function, $f : \boldsymbol{x_j^{(i)}} \mapsto \mathbb{R}$. At any point in the training, the model outputs the score $\boldsymbol{z^{(i)}} = \left[f\left(\boldsymbol{x_1^{(i)}}\right), \cdots, f\left(\boldsymbol{x_n^{(i)}}\right)\right]^T \in \mathbb{R}^n$. We direct readers to Appendix A.2 for detailed specifics about the L2R model, including our methods for bootstrapping it with mutual information gain and subsequent training procedures.

2.2 Leveraging L2R as Pretrained Attention

Pretrained and Fine-tuned AWD-LSTM: We use the AWD-LSTM architecture (Merity et al., 2017) as LM in our experiments[3]. That means, AWD-LSTM model learns hidden features from a sequence of n tokens $\langle t_1, t_2, \cdots, t_n\rangle$, where each token is represented by word embedding $\boldsymbol{e}_{t_j} \in \mathbb{R}^{s_e}$. The hidden feature learned by AWD-LSTM corresponding to the j^{th} token is represented as:

$$\boldsymbol{h_j} = \text{AWD-LSTM}(\langle \boldsymbol{e}_{t_1}, \cdots, \boldsymbol{e}_{t_j}\rangle), \boldsymbol{h_j} \in \mathbb{R}^{s_e} \tag{2}$$

Note that all the pretrained word embeddings $\boldsymbol{e}_{t_j}$ and the parameters of the AWD-LSTM model are finetuned on the target task using the mechanisms proposed in Howard and Ruder (2018).

[3]We used the pretrained LM from https://docs.fast.ai/text.models.awdlstm.html

Decoder – PLANT L2R as Attention: To allocate label-specific attention weights to the most informative tokens in the sequence $\langle t_1, t_2, \cdots, t_n\rangle$ we take the following three steps.

First, the hidden features $\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n$ of the sequence $\langle t_1, t_2, \cdots, t_n\rangle$ are concatenated to formulate the matrix $\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n]^T \in \mathbb{R}^{n \times s_e}$. To transform $\boldsymbol{H}$ into label-specific vectors, we compute label-specfic attention weights as:

$$\boldsymbol{A} = \text{softmax}(\boldsymbol{H}\boldsymbol{U}^T), \boldsymbol{A} \in \mathbb{R}^{n \times |\mathcal{L}|} \tag{3}$$

where $\boldsymbol{U} \in \mathbb{R}^{|\mathcal{L}| \times s_e}$ is the label embedding matrix.The i^{th} column in $\boldsymbol{A}$ represents the attention weights corresponding to the i^{th} label in $\mathcal{L}$ for each of the n tokens. To ensure the bulk of the weight is placed on the most informative tokens, the softmax is applied at the column level. Here $\boldsymbol{A}$ denotes the *learned* attention weights.

Second, we perform attention planting by utilizing two types of attention weights: *static-planted* ($\boldsymbol{S}$) and *differentiable-planted* ($\boldsymbol{P}$). The static-planted attention ($\boldsymbol{S}$) remains constant and is based on mutual information gain, while the differentiable-planted attention ($\boldsymbol{P}$) comprises trainable parameters. These mechanisms enhance the model's ability to prioritize relevant tokens. We determine the static-planted attention as $\boldsymbol{S} = \left[\boldsymbol{g}^{(1)}, \boldsymbol{g}^{(2)}, \cdots, \boldsymbol{g}^{(|\mathcal{L}|)}\right] \in \mathbb{R}^{n \times |\mathcal{L}|}$, is comprised of individual vectors $\boldsymbol{g^{(i)}} = \left[g_1^{(i)}, g_2^{(i)}, \cdots, g_n^{(i)}\right]^T \in \mathbb{R}^n$. Each element $g_j^{(i)}$ of these vectors represents the relevance of token t_j with respect to label $l^{(i)}$, as precisely defined in section 2.1. We determine the differentiable-planted attention by computing feature vectors $\boldsymbol{x_j^{(i)}} = \Psi\left(\boldsymbol{e}_{l^{(i)}}, \boldsymbol{e}_{t_j}\right)$ for each label-token pair $\left(l^{(i)}, t_j\right), i = 1, 2, \cdots, |\mathcal{L}|; j = 1, 2, \cdots, n$ as per equation 1. Then utilizing pretrained embeddings $\boldsymbol{e}_{l^{(i)}}$ and $\boldsymbol{e}_{t_j}$ from the L2R model in section 2.1, the pretrained L2R model computes scores $\boldsymbol{P} = \left[\boldsymbol{p}^{(1)}, \boldsymbol{p}^{(2)}, \cdots, \boldsymbol{p}^{(|\mathcal{L}|)}\right] \in \mathbb{R}^{n \times |\mathcal{L}|}$, where $\boldsymbol{p^{(i)}} = \left[f\left(\boldsymbol{x_1^{(i)}}\right), \cdots, f\left(\boldsymbol{x_n^{(i)}}\right)\right]^T \in \mathbb{R}^n$, and f is the ranking function from equation 7. In a departure from the standard attention approach, we introduce *inattention*, a pre-softmax thresholding technique that strategically elevates the significance of attention weights. By effectively zeroing out less relevant tokens, this method ensures maximal focus

on pivotal tokens:

$$\boldsymbol{P} = \mathsf{softmax}(\mathsf{threshold}(\boldsymbol{P}, k)) \tag{4}$$

where both threshold (Appendix A.3) and softmax are applied at the column level.

Third, to compute the label-specific vectors, we perform linear combinations of the hidden features $\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_n$ using the attention weights from three sources: the *learned* attention weights in each column of $\boldsymbol{A}$, the *static-planted* attention weights in each column of $\boldsymbol{S}$, and the *differentiable-planted* attention weights in each column of $\boldsymbol{P}$. This is followed by element-wise matrix multiplication with a weight matrix $\boldsymbol{W} \in \mathbb{R}^{|\mathcal{L}| \times s_e}$:

$$\boldsymbol{V} = (\boldsymbol{A}^T\boldsymbol{H} + \boldsymbol{S}^T\boldsymbol{H} + \boldsymbol{P}^T\boldsymbol{H}) \odot \boldsymbol{W}, \boldsymbol{V} \in \mathbb{R}^{|\mathcal{L}| \times s_e} \tag{5}$$

The purpose of $\boldsymbol{W}$ is to boost attention. The i^{th} row $\boldsymbol{v}_i$ of $\boldsymbol{V}$, can be thought of as the information regarding the i^{th} label captured by *attention* from the token sequence $\langle t_1, t_2, \cdots, t_n \rangle$. Finally, this label-specific information is summed and added with a label-specific bias followed by sigmoid activation to produce predictions:

$$\hat{\boldsymbol{y}} = \mathsf{sigmoid}(\mathbf{1}\boldsymbol{V}^T + \boldsymbol{b}); \mathbf{1} \in \mathbb{R}^{s_e}; \boldsymbol{b}, \hat{\boldsymbol{y}} \in \mathbb{R}^{|\mathcal{L}|} \tag{6}$$

The training objective is to mimimize the binary cross-entropy loss between $\hat{\boldsymbol{y}}$ and the target $\boldsymbol{y}$ as:

$$\text{Loss}(\boldsymbol{y}, \hat{\boldsymbol{y}}, \theta) = \sum_{i=1}^{|\mathcal{L}|} y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i),$$

where θ denotes all trainable model parameters.

Inattention: In contrast to traditional attention mechanisms, we introduce *inattention* (Equation 4) a novel technique that strategically enhances attention weights' significance by filtering less relevant tokens, ensuring focus on critical elements within a token sequence. Our ablation analysis consistently identifies the optimal threshold parameter k in the range $[1, 10k']$, where k' is from the nDCG@k loss function (Equation 8) for L2R model pretraining (Section 2.1). This aligns with our motivation to use an L2R model to learn token ranks, concentrating attention on informative tokens with higher ranks while reducing attention to less relevant tokens.

Stateful Decoder: Our decoder innovates with a *stateful* mechanism inspired by backpropagation through time (BPTT) (Howard and Ruder, 2018). Segmentation into fixed-size batches preserves the *state*, consisting of the last hidden feature $\boldsymbol{h}_n$ and prediction $\hat{\boldsymbol{y}}_b$ for each batch. This state guides subsequent batches, allowing cumulative predictions through initializing the AWD-LSTM encoder with $\boldsymbol{h}_n$ and continuously adding predictions. Gradients propagate back across batches, improving adaptability to large documents and model convergence. Our stateful decoder eliminates the need for text truncation (Li and Yu, 2020; Xie et al., 2019), preventing performance loss, and ensures stable GPU memory usage by processing long texts in manageable batches.

Discriminative Fine-tuning and Gradual Unfreezing: To fine-tune our pretrained model effectively for attention planting, we employ two essential strategies. First, we leverage *discriminative fine-tuning* (Howard and Ruder, 2018). This technique assigns distinct learning rates (LR) to different parameter groups $\theta^l \in \left\{\theta^e, \theta^p, \theta^d\right\}$ corresponding to AWD-LSTM encoder, planted decoder, and the remaining model components. This approach optimizes the pretrained model by focusing on areas that need the most adjustment. The update rule for discriminative fine-tuning is as follows:

$$\theta_t^l = \theta_{t-1}^l - \eta^l \cdot \nabla_{\theta^l} J(\theta)$$

where $\nabla_{\theta^l} J(\theta)$ is the gradient with respect to the model's loss function. For experiments we applied half or a third of the LR for already proficient L2R model parameters compared to others. Second, we embrace *gradual unfreezing* (Howard and Ruder, 2018). This method fine-tunes the model in a layer-wise sequence, starting from the last layer and moving gradually towards the initial layers.

Bidirectional Language Model: For the `MIMIC-III-full` and `MIMIC-IV-full` (Table 2), we pretrain both a forward and backward LM. We fine-tune an XMTC model for each LM independently and average the classifier predictions. On `MIMIC-III-full` P@15 increased from 60.61 to 61.67, and on `MIMIC-IV-full`, from 54.5 to 55.6.

3 Experiments

3.1 Experimental Setup

Datasets: We compare PLANT to SOTA ICD coding models (Yang et al., 2022; Zhang et al., 2022; Yuan et al., 2022; Liu et al., 2021; Vu et al., 2021; Li and Yu, 2020; Cao et al., 2020; Xie et al., 2019; Mullenbach et al., 2018). Our primary datasets are MIMIC-III (Johnson et al., 2016)

and the newly available MIMIC-IV (Johnson et al., 2023). These datasets contain rich textual and structured records from ICU settings, with a focus on discharge summaries. These summaries are meticulously annotated with ICD-9 codes (MIMIC-III) and ICD-10 codes (MIMIC-IV) to represent diagnoses and procedures. MIMIC-III comprises 52,722 discharge summaries and 8,929 unique ICD-9 codes. We follow the methodology in (Mullenbach et al., 2018), including patient ID-based splits for full-code experiments and a subset of 50 frequent codes. We also evaluate our model on the few-shot MIMIC-III-rare50 dataset (Yang et al., 2022), featuring 50 rare ICD codes. Additionally, we explore MIMIC-IV, with 122,279 discharge summaries and 7,942 unique ICD-10 codes, following Edin et al. (2023). We denote these datasets as `MIMIC-III-full`, `MIMIC-III-top50`, `MIMIC-III-rare50`, and `MIMIC-IV-full`. Refer to Table 2 for dataset statistics.

	`MIMIC-III-full`	`MIMIC-IV-full`
Number of documents	52,723	122,279
Number of patients	41,126	65,659
Number of unique codes	8,929	7,942
Codes pr. instance: Median (IQR)	$14(10-20)$	$14(9-20)$
Words pr. document: Median (IQR)	$1,375(965-1,900)$	$1,492(1,147-1,931)$
Documents: Train/val/test [%]	90.5/3.1/6.4	72.9/10.9/16.2

Table 2: Descriptive statistics for `MIMIC-III-full` and `MIMIC-IV-full` discharge summary training sets.

Preprocessing, Implementation and Hyperparameters: We direct readers to Appendix A.4 and Appendix A.5 for specifications.

Evaluation metrics: To compare with prior ICD studies, we use various metrics, focusing on micro and macro F1 scores, AUC, and P@k. Micro-averaging treats each (text, code) pair individually, while macro-averaging computes metrics per label. Micro-R reflects the ratio of true positives to the sum of true positives and false negatives for each label, while Macro-R represents the average recall across all labels. Precision follows a similar calculation pattern. Macro-averaged metrics prioritize infrequent labels. P@k denotes the proportion of the k top-scored labels that match the ground truth.

Baselines: These included models such as CAML (Mullenbach et al., 2018), MSATT-KG (Xie et al., 2019), MUltiResCNN (Li and Yu, 2020), HyperCore (Cao et al., 2020), LAAT/JointLAAT (Vu et al., 2021), ISD (Zhou et al., 2021), Effective-CAN (Liu et al., 2021), MSMN (Yuan et al., 2022), DiscNet (Zhang et al., 2022), and KEPTLongformer (Yang et al., 2022).

3.2 Main Results

`MIMIC-III-full` (Table 3): PLANT demonstrated notable enhancements over existing SOTA models. Specifically, when compared with Effective-CAN, LAAT, and DiscNet, PLANT yielded superior performance in terms of micro-F1, P@5, P@8, and P@15 metrics, with improvements of 0.5%, 2.7%, 0.6%, and 0.3%, respectively. Significantly, PLANT achieved a remarkable P@5 score of 84%, indicative of an average of 4.2 correct predictions among the top 5; while demonstrating only a slightly lower micro-AUCthan DiscNet.

Model	AUC		F1		P@k		
	Macro	Micro	Macro	Micro	P@5	P@8	P@15
CAML/DR-CAML	89.7	98.6	8.8	53.9	-	70.9	56.1
MSATT-KG	91.0	99.2	9.0	55.3	-	72.8	58.1
MultiResCNN	91.0	98.6	8.5	55.2	-	73.4	58.4
HyperCore	93.0	98.9	9.0	55.1	-	72.2	57.9
LAAT/JointLAAT	92.1	98.8	10.7	57.5	81.3	73.8	59.1
ISD	93.8	99.0	11.9	55.9	-	74.5	-
Effective-CAN	92.1	98.9	10.6	58.9	-	75.8	60.6
MSMN	95.0	99.2	10.3	58.4	-	75.2	59.9
DiscNet	**95.6**	**99.3**	**14.0**	58.8	-	76.5	61.4
PLANT (Ours)	90.4	98.9	10.1	**59.4***	**84.0***	**77.1***	**61.7***

Table 3: Results (in %) on the `MIMIC-III-full` test set. We ran our model 5 times each with different random seeds for initialization and report mean scores. * indicates that the performance difference between PLANT and the next best is significant ($p < 0.01$, using the Approximate Randomization test). All scores in tables 3, 4, 5 and 6 are reported under the same experimental setup.

`MIMIC-III-top50` (Table 4): PLANT outperformed the previous SOTA baseline models of MSMN and LAAT with regard to macro-F1, micro-F1, P@8 and P@15, respectively; while matching micro-AUCwith ISD and achieving a slightly lower P@5 as compared to MSMN. PLANT produced improvements of 0.4%, 0.3%, 0.3% and 1.4% for macro-F1, micro-F1, P@8 and P@15, respectively.

Model	AUC		F1		P@k		
	Macro	Micro	Macro	Micro	P@5	P@8	P@15
CAML/DR-CAML	88.4	91.6	57.6	63.3	61.8	-	-
MSATT-KG	91.4	93.6	63.8	68.4	64.4	-	-
MultiResCNN	89.9	92.8	60.6	67.0	64.1	-	-
HyperCore	89.5	92.9	60.9	66.3	63.2	-	-
LAAT/JointLAAT	92.5	94.6	66.6	71.6	67.5	54.7	35.7
ISD	**93.5**	**94.9**	67.9	71.7	68.2	-	-
Effective-CAN	92.0	94.5	66.8	71.7	66.4	-	-
MSMN	92.8	94.7	68.3	72.5	**68.0**	-	-
PLANT (Ours)	93.1	**94.9**	**68.7**	**72.8**	67.2	**55.0***	**36.3***

Table 4: Results on the `MIMIC-III-top50` test set.

`MIMIC-III-rare50` (Table 5): PLANT surpassed the prior SOTA baseline, KEPTLongformer, by astounding margins. Specifically, by 12.9% in macro-AUC, 12.7% in micro-AUC, 52.2% in

Model	AUC		F1	
	Macro	Micro	Macro	Micro
MSMN	75.3	76.2	17.1	17.2
KEPTLongformer	82.7	83.3	30.4	32.6
PLANT (Ours)	**95.6***	**96.0***	**82.6***	**84.2***

Table 5: Results on the `MIMIC-III-rare50` test set.

macro-F1, and 51.6% in micro-F1. Intriguingly, it's worth noting that these remarkable results were achieved by training with only unfrozen PLANT layers, without even utilizing the entire model's capacity. This underscores the extraordinary potential of PLANT in delivering outstanding performance with efficient training strategies in low-shot settings.

`MIMIC-IV-full` (Table 6): PLANT outperformed previous SOTA baseline model of LAAT with regard to P@8 and P@15 while matching micro-AUCwith LAAT. PLANT produced improvements of 1.7%, 1.3% for P@8 and P@15, respectively.

Model	AUC		F1		P@k		
	Macro	Micro	Macro	Micro	P@5	P@8	P@15
CAML/DR-CAML	91.1	98.5	16.0	55.4	-	66.8	52.2
MultiResCNN	94.5	**99.0**	21.1	56.9	-	67.8	53.5
LAAT/JointLAAT	**95.4**	**99.0**	20.3	57.9	-	68.9	54.3
PLANT (Ours)	94.8	**99.0**	19.6	57.1	**78.1***	**70.6***	**55.6***

Table 6: Results on the `MIMIC-IV-full` test set. The comparitive results are reported from Edin et al. (2023).

4 Analysis

Firstly, except for the Gradual Unfreezing and Bidirectionality, we selectively unfreeze the layers in decoder, keeping the encoder frozen—meaning no backpropagation was performed on their weights during training. This ensures that performance improvements are attributed directly to the decoder, our primary focus. Secondly, all reported performance metrics stem from the full test sets of both `MIMIC-III-full` and `MIMIC-IV-full` datasets. Thirdly, reported enhancements were statistically significant ($p < 0.01$, using the Approximate Randomization test).

Impact of PLANT (Figure 3,4): We evaluate PLANT and LAAT (Vu et al., 2021) in contexts with skewed label distributions. PLANT uses pretrained L2R activations $\boldsymbol{P}$ and mutual information gain $\boldsymbol{S}$, initializing the decoder's attention weights. While LAAT relies solely on learned attention $\boldsymbol{A}$, initialized randomly and learned from scratch. That is LANT omits $\boldsymbol{P}$ and $\boldsymbol{S}$ form Equation 5. Our analysis involves training both PLANT and LAAT models across varying fractions of a balanced training dataset, with both models trained for up to five epochs. The test set remains constant, and we measure P@5 and P@15 as the performance metric for both models. The results were notable: the PLANT model consistently matched or surpassed the LAAT model's performance across all training sizes, even with significantly less data. For instance, in the case of `MIMIC-IV-full`, PLANT achieved a P@5 of 0.50 and P@15 of 0.37 with a smaller training split of 1090 and 2743 instances, respectively, matching the performance of the LAAT model trained on a significantly larger split of 10,337 and 12,902 instances. Similarly, in the case of `MIMIC-III-full`, PLANT achieved a P@5 of 0.47 and P@15 of 0.30, trained with only 136 and 235 instances, respectively. This performance equates to that of the LAAT model trained on a dataset comprising 1342 and 1578 instances. These findings are visually represented in Figure 3 through vertical and horizontal lines, illustrating the substantial efficiency gains of PLANT in terms of training data requirements while maintaining or improving model performance. Since PLANT achieves comparable performance to LAAT with significantly less data, which also implies a lower number of instances per label (aka skewed label distribution), this outcome underscores the inefficiencies of the LAAT approach in such scenarios. To examine overfitting (Figure 4), we trained both PLANT and LAAT on `MIMIC-IV-full` for 60 epochs. While PLANT remained stable, LAAT began overfitting after 40 epochs, diverging train and test loss, leading to a decline in P@15.

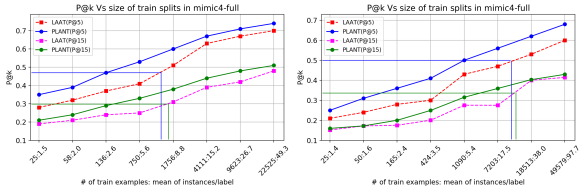

Figure 3: P@15 for PLANT vs. LAAT (Vu et al., 2021) with different number of training examples on `MIMIC-III-full` and `MIMIC-IV-full`.

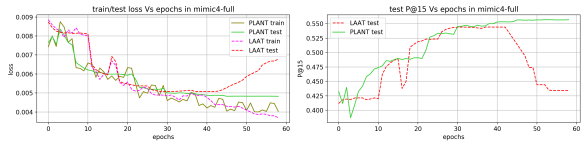

Figure 4: PLANT does not overfit on `MIMIC-IV-full`, LAAT (Vu et al., 2021) does.

Ablation	MIMIC-III-full	MIMIC-IV-full
Without Inattention	50.95	42.40
With Inattention	51.05	42.51
Stateless	52.80	43.38
Stateful	52.90	44.22
− disc	51.40	43.29
+ disc	52.21	44.34
full unfreezing	57.78	49.78
gradual unfreezing	58.31	50.97

Table 7: P@15 for MIMIC-III-full and MIMIC-IV-full (train split 49,579) test set.

Impact of Inattention (Table 7): We investigated the impact of the inattention threshold k (Equation 4) within PLANT on MIMIC-III-full and MIMIC-IV-full. The training splits comprised 22,525 instances (average 49 instances per label) and 49,579 instances (average 97 instances per label) for the respective datasets. We trained each model for 5 epoch and measured P@15. For MIMIC-III-full, the model without inattention ($k = 72$) achieved a P@15 of 50.95, while the model with inattention ($k = 56$) achieved a slightly higher P@15 of 51.05. In the case of MIMIC-IV-full, the model without inattention attained a P@15 of 42.4, which improved to 42.51 with inattention ($k = 8$).

Impact of Sateful Decoder (Table 7): On the MIMIC-III-full training dataset, using the stateful decoder for three epochs yielded a P@15 of 52.9, a slight improvement over 52.8 without it. Similarly, on the MIMIC-IV-full (training split of 49,579), employing the stateful decoder for seven epochs significantly boosted P@15, from 43.28 to 44.22. These improvements highlight the stateful decoder's role in enhancing PLANT's performance with extensive text data.

Impact of Discriminative Fine-tuning and Gradual Unfreezing (GU) (Table 7): On the MIMIC-III-full, training PLANT for one epoch with discriminative fine-tuning, applying half the learning rate to L2R parameters, improved P@15 from 51.40 to 52.21 on the test set. Similarly, on MIMIC-IV-full (training split of 49,579), training PLANT for seven epochs with a third of the learning rate for L2R parameters increased P@15 from 43.29 to 44.34. For GU we explored two scenarios: one gradually unfreezing the model layer by layer, and the other unfreezing the entire model simultaneously. Both models were trained for 10 epochs. On the MIMIC-III-full, GU increased P@15 from 57.78 to 58.31; and on MIMIC-IV-full from 49.78 to 50.97.

518.81: Acute respiratory failure
PLANT: ...patient had a gcs3t and required **intubation** ...**fio2** ··· temp po2 **pco2** ph ...
MSATT-KG: ... left **hemothorax**, **ETOH**, **depression**, stable discharge condition...
CAML: ...**small apical pneumothorax remained unchanged ... now tolerating a** ...
Text-CNN: ...**revealed a persistent left pleural effusion and due** to concern for loculated hemothorax...

530.81: Esophageal reflux
PLANT: ... **gastroesophageal reflux** ... home o2 **gerd** osteoporosis ... one puff hospital1 **prilosec** 20mg....
MSATT-KG: ... **tracheostomy & feeding gastrostomy** ... GERD, anxiety ...
CAML: ... **rib fx requiring tracheostomy & feeding gastrostomy,**, GERD, anxiety, cataracts...
Text-CNN: ... **right thoracotomy, decortication of lung, mobilization of liver** off of chest wall...

Table 8: Interpretability evaluation results for different models.

Interpretability Case Study (Table 8): We compare PLANT's interpretability against three baselines: MSATT-KG, CAML, and Text-CNN(Kim, 2014). While PLANT selects top 5 tokens per label based on attention values, baseline methods extract informative n-grams. MSATT-KG employs multi-scale and label-dependent attention, while CAML and Text-CNN use label-dependent attention and different phrase selection strategies. CAML uses a receptive field, and Text-CNN selects positions based on maximum channel values. In the interpretability case study, PLANT attends to tokens like 'intubation', 'fio2', and 'pc02'. 'fio2' represents Fraction of Inspired Oxygen, critical in determining oxygen concentration delivered to a patient. 'PCO2' signifies partial pressure of carbon dioxide, indicative of conditions like respiratory acidosis or alkalosis. In another example, informative tokens include 'gastrophageal', 'reflux', 'gerd', and 'prilosec', where 'gerd' denotes Gastroesophageal Reflux Disease and 'prilosec' is a proton pump inhibitor.

5 Related Work: Automatic ICD Coding

Xie and Xing (2018) introduced LSTM with tree structures and adversarial learning, Prakash et al. (2017) utilized condensed memory neural networks on MIMIC-III (Johnson et al., 2016). Baumel et al. (2017) proposed a hierarchical GRU network. Further enhancements include Xie et al. (2019)'s convolutions and multi-scale feature attention, Li and Yu (2020)'s convolutional layers, and Cao et al. (2020)'s graph convolution and hyperbolic representation. Vu et al. (2021) introduced LSTM-based attention models, Zhou et al. (2021) proposed shared representation networks, Liu et al. (2021) improved convolutional networks, and Yuan et al. (2022) introduced multi-synonyms attention networks. Zhang et al. (2022) addressed discourse structure and code-description reconciliation, including physician informal abbreviations.

Limitations

The PLANT method, while effective, presents a notable trade-off in terms of computational resources. The necessity to pretrain and load the L2R model imposes a substantial memory overhead compared to traditional attention mechanisms. Consequently, our memory constraints limited the number of epochs for which PLANT could be trained. This aspect of PLANT, particularly its scalability to larger XMTC datasets, warrants further investigation. Future work will explore strategies to optimize memory usage, ensuring that the benefits of PLANT can be harnessed more broadly without the current limitations on training duration and dataset size.

Broader Impacts and Ethical Considerations

Our research contributes to the broader field of natural language processing (NLP) and machine learning (ML), advancing the SOTA in XMTC. In the context of XMTC, our research has the potential to significantly impact various sectors, including healthcare, finance, and e-commerce. By automating labor-intensive tasks such as medical coding and diagnosis, these models can enhance healthcare accessibility, particularly in underserved communities. This can lead to improved patient outcomes and reduced disparities in healthcare access. Additionally, in education, XMTC models can support personalized learning experiences by categorizing educational resources and recommending tailored learning materials to students. Furthermore, XMTC can contribute to policy development by analyzing public opinion and sentiment from social media and news sources, providing valuable insights for policymakers to develop evidence-based policies and interventions. These applications demonstrate the diverse and far-reaching societal implications of XMTC technology. However, we acknowledge the importance of ensuring that automated systems do not perpetuate biases or discrimination present in the data. Therefore, we prioritize fairness, transparency, and accountability in our model development process. In summary, while our research presents exciting opportunities for automation and efficiency gains, we recognize the importance of ethical considerations and broader societal impacts. By upholding ethical principles and promoting responsible AI development, we aim to maximize the positive impact of our work while mitigating potential risks.

References

Tal Baumel, Jumana Nassour-Kassis, Raphael Cohen, Michael Elhadad, and Noémie Elhadad. 2017. Multi-label classification of patient notes a case study on icd code assignment. *arXiv preprint arXiv:1709.09587*.

Alex Bottle and Paul Aylin. 2008. Intelligent information: a national system for monitoring clinical performance. *Health services research*, 43(1p1):10–31.

Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81.

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, Online. Association for Computational Linguistics.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2018. Universal transformers. *arXiv preprint arXiv:1807.03819*.

Joakim Edin, Alexander Junge, Jakob D Havtorn, Lasse Borgholt, Maria Maistro, Tuukka Ruotsalo, and Lars Maaløe. 2023. Automated medical coding on mimic-iii and mimic-iv: A critical review and replicability study. *arXiv preprint arXiv:2304.10909*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Alistair EW Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, et al. 2023. Mimic-iv, a freely accessible electronic health record dataset. *Scientific data*, 10(1):1.

Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, 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.

Douwe Kiela, Changhan Wang, and Kyunghyun Cho. 2018. Dynamic meta-embeddings for improved sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1477, Brussels, Belgium. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Fei Li and Hong Yu. 2020. Icd coding from clinical text using multi-filter residual convolutional neural

network. In *proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8180–8187.

Yang Liu, Hua Cheng, Russell Klopfer, Matthew R. Gormley, and Thomas Schaaf. 2021. Effective convolutional attention network for multi-label clinical document classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5941–5953, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and optimizing lstm language models. *arXiv preprint arXiv:1708.02182*.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. 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, Volume 1 (Long Papers)*, pages 1101–1111, New Orleans, Louisiana. Association for Computational Linguistics.

Anthony N Nguyen, Donna Truran, Madonna Kemp, Bevan Koopman, David Conlan, John O'Dwyer, Ming Zhang, Sarvnaz Karimi, Hamed Hassanzadeh, Michael J Lawley, et al. 2018. Computer-assisted diagnostic coding: effectiveness of an nlp-based approach using snomed ct to icd-10 mappings. In *AMIA Annual Symposium Proceedings*, volume 2018, page 807. American Medical Informatics Association.

Kimberly J O'malley, Karon F Cook, Matt D Price, Kimberly Raiford Wildes, John F Hurdle, and Carol M Ashton. 2005. Measuring diagnoses: Icd code accuracy. *Health services research*, 40(5p2):1620–1639.

Aaditya Prakash, Siyuan Zhao, Sadid Hasan, Vivek Datla, Kathy Lee, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2017. Condensed memory networks for clinical diagnostic inferencing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018. Why self-attention? a targeted evaluation of neural machine translation architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Thanh Vu, Dat Quoc Nguyen, and Anthony Nguyen. 2021. A label attention model for icd coding from clinical text. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online. Association for Computational Linguistics.

Pengtao Xie and Eric Xing. 2018. A neural architecture for automated icd coding. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1066–1076.

Xiancheng Xie, Yun Xiong, Philip S. Yu, and Yangyong 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*, CIKM '19, page 649–658, New York, NY, USA. Association for Computing Machinery.

Song Xu, Haoran Li, Peng Yuan, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. Self-attention guided copy mechanism for abstractive summarization. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 1355–1362.

Zhichao Yang, Shufan Wang, Bhanu Pratap Singh Rawat, Avijit Mitra, and Hong Yu. 2022. Knowledge injected prompt based fine-tuning for multi-label few-shot icd coding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2022, page 1767. NIH Public Access.

Zheng Yuan, Chuanqi Tan, and Songfang Huang. 2022. Code synonyms do matter: Multiple synonyms matching network for automatic ICD coding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 808–814, Dublin, Ireland. Association for Computational Linguistics.

Shurui Zhang, Bozheng Zhang, Fuxin Zhang, Bo Sang, and Wanchun Yang. 2022. Automatic ICD coding exploiting discourse structure and reconciled code embeddings. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2883–2891, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Tong Zhou, Pengfei Cao, Yubo Chen, Kang Liu, Jun Zhao, Kun Niu, Weifeng Chong, and Shengping Liu. 2021. Automatic ICD coding via interactive shared representation networks with self-distillation mechanism. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5948–5957, Online. Association for Computational Linguistics.

A Appendix

A.1 Skewness of Codes

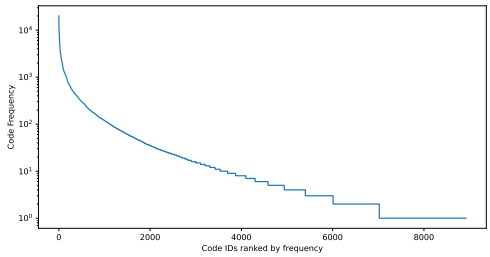

Figure 5: The skewness of ICD-9-CM code distribution for MIMIC-III (Johnson et al., 2016).

A.2 L2R Model (continued from Section 2.1)

The ranking function, $f : \boldsymbol{x_j^{(i)}} \mapsto \mathbb{R}$, of the L2R model is an L layered feed forward network,

$$f(\boldsymbol{x_j^{(i)}}) = y^L, y^{(l)} = a(W^{(l)} \cdot y^{(l-1)} + b^{(l)}), \quad (7)$$

where $y^{(l)}$ is layer l output, $y^{(0)} = x$ is input, $W^{(l)}$ is layer l weight matrix, $b^{(l)}$ is layer l bias vector, and $a(\cdot)$ is the activation function. In our experiments we trained the L2R model with $L = 2$.

At any point in the training, the model outputs the score $\boldsymbol{z^{(i)}} = \left[f\left(\boldsymbol{x_1^{(i)}}\right), \cdots, f\left(\boldsymbol{x_n^{(i)}}\right)\right]^T \in \mathbb{R}^n$. The objective of the L2R model is to minimize the total loss,

$$\sum_{i=1}^{m} \mathsf{nDCG@k}\left(\boldsymbol{z^{(i)}}, \boldsymbol{g^{(i)}}\right), \quad (8)$$

where nDCG@k is the maximum allowable DCG@k, which is defined as:

$$\mathsf{DCG@k}\left(\boldsymbol{z^{(i)}}, \boldsymbol{g^{(i)}}\right) := \sum_{l \in \mathsf{rank}_k(\boldsymbol{z^{(i)}})} \frac{2^{g_l^{(i)}}}{\log(l+1)}.$$ [4]

Bootstrapping L2R Model: Let (I, J) be a pair of random variables for the label $l^{(i)}$ and token t_j over the space $\mathcal{I} \times \mathcal{J}$, where $\mathcal{I} = \{\text{label } i \text{ present}, \text{label } i \text{ not present}\}$ and $\mathcal{J} = \{\text{token } j \text{ present}, \text{token } j \text{ not present}\}$. Then, g_j^i is defined as the mutual information gain of I and J:

$$g_j^{(i)} = \sum_{x \in \mathcal{I}, y \in \mathcal{J}} P_{(I,J)}(x, y) \log\left(\frac{P_{(I,J)}(x, y)}{P_I(x) P_J(y)}\right),$$

where $P_{(I,J)}$ is the joint, and P_I, P_J are the marginal probability mass function of I and J, respectively.

Training L2R Model: Gradient update rule to train the L2R model on $\left\{\left(\boldsymbol{X^{(i)}}, \boldsymbol{g^{(i)}}\right)\right\}_{i=1}^{m}$ are defined as follows. Let $I^{(i)}$ denote the set of pairs of token indices $\{j, k\}$, such that $g_j^{(i)} > g_k^{(i)}$. Also, let $z_j^{(i)} = f\left(\boldsymbol{x_j^{(i)}}\right)$ and $z_k^{(i)} = f\left(\boldsymbol{x_k^{(i)}}\right)$. The parameters of L2R model, $w_p \in \mathbb{R}$, are updated as (Burges, 2010):

$$\delta w_p = -\eta \sum_j \lambda_j \frac{\partial z_j^{(i)}}{\partial w_k},$$

$$\lambda_j = \sum_{k:\{j,k\} \in I^{(i)}} \lambda_{jk} - \sum_{k:\{k,j\} \in I^{(i)}} \lambda_{kj},$$

$$\lambda_{jk} = -\frac{\sigma}{1 + e^{\sigma\left(z_j^{(i)} - z_k^{(i)}\right)}} |\Delta \mathsf{nDCG@k}|_{jk},$$

where $|\Delta \mathsf{nDCG@k}|_{jk}$ denotes the change in nDCG@k by swapping j and k in $\mathsf{rank}(\boldsymbol{z^{(i)}})$.

A.3 Threshold

$$\mathsf{threshold}(\boldsymbol{p}^i, k) = \begin{cases} p_j, & \text{if } p_j > k^{th} \text{ largest } p \\ 0 & \text{otherwise.} \end{cases}$$

A.4 Preprocessing

Following prior research (Mullenbach et al., 2018; Xie et al., 2019; Li and Yu, 2020), we tokenize and lowercase all text while eliminating non-alphabetic tokens containing numbers or punctuation. A distinctive feature of our approach is the absence of preprocessed word embeddings. Instead, we fine-tune a pretrained AWD-LSTM model on our target dataset, allowing for parameter refinement, including word embeddings, and the generation of context-specific embeddings for new words in the dataset. While the concept of fine-tuning pretrained models is not new (Howard and Ruder, 2018), our innovation lies in its application to the XMTC domain. Contrary to previous practices (Li and Yu, 2020), we refrain from truncating text, as our experiments and findings align with those of Zhang et al. (2022), which demonstates substantial performance variation due to truncation. To handle longer texts, we employ our stateful decoder (refer to Section 2.2).

A.5 Implementation and Hyperparameters

We ensure robustness across diverse XMTC datasets by fine-tuning hyperparameters on the

[4] here $\mathsf{rank}_k(\boldsymbol{z^{(i)}})$ returns the k largest indices of $\boldsymbol{g^{(i)}}$ ranked in descending order.

MIMIC-III-full and MIMIC-IV-full validation sets. Experiments are conducted on an NVIDIA QUADRO RTX 8000 GPU with 48 GB VRAM. We utilize the AWD-LSTM LM with an embedding size of 400, 3 LSTM layers with 1152 hidden activations, and the Adam Optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and weight decay of 0.01. During fine-tuning, we apply dropout rates and weight dropout, with a batch size of 384, BPTT of 80, 20 epochs, and a learning rate of $1e - 5$. Classifier training also includes dropout rates and weight dropout, with a batch size of 16, BPTT of 72, and discriminative fine-tuning with gradual unfreezing over 115 epochs (on MIMIC-III-full), alongside scheduled weight decay and learning rate ranges.