HyperCore: Hyperbolic and Co-graph Representation for Automatic ICD Coding

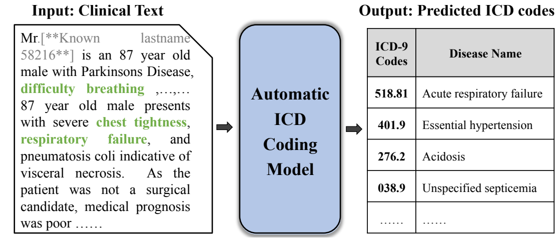
Pengfei Cao[1,2], **Yubo Chen**[1,2], **Kang Liu**[1,2], **Jun Zhao**[1,2],
Shengping Liu[3] and **Weifeng Chong**[3]
[1] National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, 100190, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences,
Beijing, 100049, China
[3] Beijing Unisound Information Technology Co., Ltd, Beijing, 100028, China
{pengfei.cao, yubo.chen, kliu, jzhao}@nlpr.ia.ac.cn,
{liushengping, chongweifeng}@unisound.com

Abstract

The International Classification of Diseases (ICD) provides a standardized way for classifying diseases, which endows each disease with a unique code. ICD coding aims to assign proper ICD codes to a medical record. Since manual coding is very laborious and prone to errors, many methods have been proposed for the automatic ICD coding task. However, most of existing methods independently predict each code, ignoring two important characteristics: **Code Hierarchy** and **Code Co-occurrence**. In this paper, we propose a **Hyper**bolic and **Co**-graph **Re**presentation method (HyperCore) to address the above problem. Specifically, we propose a hyperbolic representation method to leverage the code hierarchy. Moreover, we propose a graph convolutional network to utilize the code co-occurrence. Experimental results on two widely used datasets demonstrate that our proposed model outperforms previous state-of-the-art methods.

1 Introduction

The International Classification of Diseases (ICD) is a healthcare classification system supported by the World Health Organization, which provides a unique code for each disease, symptom, sign and so on. ICD codes have been widely used for analyzing clinical data and monitoring health issues (Choi et al., 2016; Avati et al., 2018). Due to the importance of ICD codes, ICD coding – which assigns proper ICD codes to a medical record – has drawn much attention. The task of ICD coding is usually undertaken by professional coders according to doctors' diagnosis descriptions in the form of free texts. However, manual coding is very expensive, time-consuming and error-prone. The cost incurred by coding errors and the financial investment spent on improving coding quality are estimated to be $25 billion per year in the US (Lang, 2007). Two main reasons can account for this. First, only the people who have medical expert knowledge and specialized ICD coding skills can handle the task. However, it is hard to train such an eligible ICD coder. Second, it is difficult to correctly assign proper codes to the input document even for professional coders, because one document can be assigned multiple ICD codes and the number of codes in the taxonomy of ICD is large. For example, there are over 15,000 and 60,000 codes respectively in the ninth version (ICD-9) and the tenth version (ICD-10) of ICD taxonomies.



Mr.[**Known lastname 58216**] is an 87 year old male with Parkinsons Disease, **difficulty breathing** ,…,… 87 year old male presents with severe **chest tightness**, **respiratory failure**, and pneumatosis coli indicative of visceral necrosis. As the patient was not a surgical candidate, medical prognosis was poor ……

ICD-9 Codes	Disease Name
518.81	Acute respiratory failure
401.9	Essential hypertension
276.2	Acidosis
038.9	Unspecified septicemia
……	……

Figure 1: An example of automatic ICD coding task. The input and output of the automatic ICD coding model are clinical text and predicted ICD codes, respectively. For better understanding, we add the corresponding disease name for each code.

To reduce human labor and coding errors, many methods have been carefully designed for automatic ICD coding (Perotte et al., 2013; Mullenbach et al., 2018). For example in Figure 1, given the clinical text of a patient, the ICD coding model needs to automatically predict the corresponding ICD codes. The automatic ICD coding task can be modeled as a multi-label classification task since each clinical text is usually accompanied by mul-

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3105–3114
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

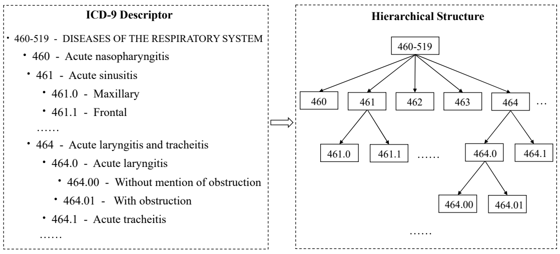


Figure 2: An example of ICD-9 descriptors and the derived hierarchical structure.

tiple codes. Most of the previous methods handle each code in isolation and convert the multi-label problem into a set of binary classification problems to predict whether each code of interest presents or not (Mullenbach et al., 2018; Rios and Kavuluru, 2018). Though effective, they ignore two important characteristics: Code Hierarchy and Code Co-occurrence, which can be leveraged to improve coding accuracy. In the following, we will introduce the two characteristics and the reasons why they are critical for the automatic ICD coding.

Code Hierarchy: Based on ICD taxonomy, ICD codes are organized under a tree-like hierarchical structure as shown in Figure 2, which indicates the parent-child and sibling relations between codes. In the hierarchical structure, the upper level nodes represent more generic disease categories and the lower level nodes represent more specific diseases. The code hierarchy can capture the mutual exclusion of some codes. If code X and Y are both children of Z (i.e., X and Y are the siblings), it is unlikely to simultaneously assign X and Y to a patient in general (Xie and Xing, 2018). For example in Figure 2, if code "464.00 (*acute laryngitis without mention of obstruction*)" is assigned to a patient, it is unlikely to assign the code "464.01 (*acute laryngitis with obstruction*)" to the patient at the same time. If automatic ICD coding models ignore such a characteristic, they are prone to giving inconsistent predictions. Thus, a challenging problem is how to model the code hierarchy and use it to capture the mutual exclusion of codes.

Code Co-occurrence: Since some diseases are concurrent or have a causal relationship with each other, their codes usually co-occur in the clinical text, such as "997.91 (*hypertension*)" and "429.9 (*heart disease*)". In this paper, we call such characteristic code co-occurrence which can capture the correlations of codes. The code co-occurrence can be utilized to correctly predict some codes which are difficult to predict by only using the clinical text itself. For example in Figure 1, the code of "*acute respiratory failure*" can be easily inferred via capturing apparent clues (i.e., the green bold words) from the text. Although there are also a few clues to infer the code of "*acidosis*", they are very obscure, let alone predict the code of "*acidosis*" by only using these obscure clues. Fortunately, there is a strong association between these two diseases: one of the main causes of "*acidosis*" is "*acute respiratory failure*". This prior knowledge can be captured via the fact that the codes of the two diseases usually co-occur in clinical texts. By considering the correlation, the automatic ICD coding model can better exploit obscure clues to predict the code of "*acidosis*". Therefore, another problem is how to leverage code co-occurrence for ICD coding.

In this paper, we propose a novel method termed as **Hyper**bolic and **Co**-graph **Re**presentation method (HyperCore) to address above problems. Since the tree-likeness properties of the hyperbolic space make it more suitable for representing symbolic data with hierarchical structures than the Euclidean space (Nickel and Kiela, 2017), we propose a hyperbolic representation learning method to learn the Code Hierarchy. Meanwhile, the graph has been proved effective in modeling data correlation and the graph convolutional network (GCN) enables to efficiently learn node representation (Kipf and Welling, 2016). Thus, we devise a code co-occurrence graph (co-graph) for capturing Code Co-occurrence and exploit the GCN to learn the code representation in the co-graph.

The contributions of this paper are threefold. **Firstly**, to our best knowledge, this is the first work to propose a hyperbolic representation method to leverage the code hierarchy for automatic ICD coding. **Secondly**, this is also the first work to utilize a GCN to exploit code co-occurrence correlation for automatic ICD coding. **Thirdly**, experiments on two widely used automatic ICD coding datasets show that our proposed model outperforms previous state-of-the-art methods.

2 Related Work

Automatic ICD Coding. Automatic ICD coding is a challenging and important task in the medical informatics community, which has been studied with traditional machine learning methods (Larkey and Croft, 1996; Perotte et al., 2013) and neural network methods (Koopman et al., 2015; Rios and Kavuluru, 2018; Yu et al., 2019). Given discharge

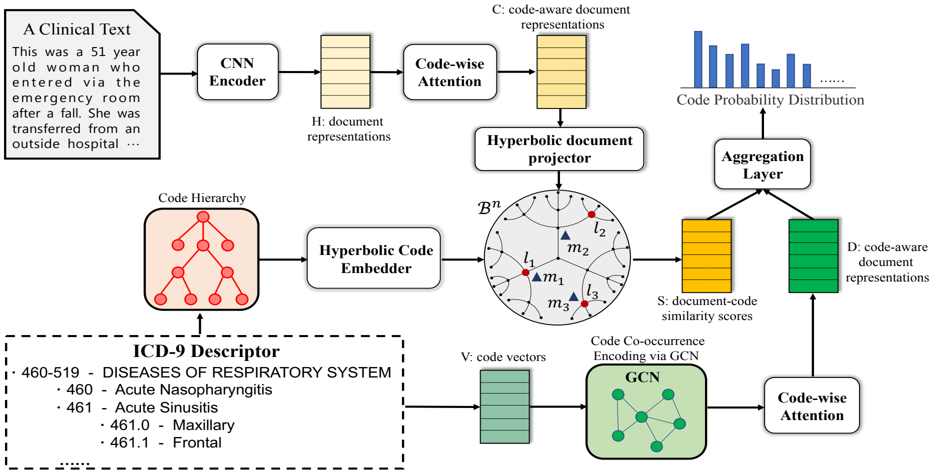

A Clinical Text
This was a 51 year old woman who entered via the emergency room after a fall. She was transferred from an outside hospital ···
CNN Encoder
H: document representations
Code-wise Attention
C: code-aware document representations
Code Probability Distribution
......
Hyperbolic document projector
Aggregation Layer
Code Hierarchy
Hyperbolic Code Embedder
$\mathcal{B}^n$
l_1
l_2
l_3
m_1
m_2
m_3
S: document-code similarity scores
D: code-aware document representations
ICD-9 Descriptor
· 460-519 - DISEASES OF RESPIRATORY SYSTEM
· 460 - Acute Nasopharyngitis
· 461 - Acute Sinusitis
· 461.0 - Maxillary
· 461.1 - Frontal
......
V: code vectors
Code Co-occurrence Encoding via GCN
GCN
Code-wise Attention

Figure 3: The architecture of **Hyper**bolic and **Co**-graph **Re**presentation method (HyperCore). In the Poincaré ball $\mathcal{B}^n$, we show the embeded code hierarchy (i.e., tree-like hierarchical structure). The dots l_i $(i = 1, 2, 3)$ on the tree-like hierarchical structure and triangles m_i $(i = 1, 2, 3)$ in the Poincaré ball denote hyperbolic code embeddings and hyperbolic document representations, respectively.

summaries, Perotte et al. (2013) propose a hierarchical SVM model to predict ICD codes. Recently, neural network methods have been introduced to the task. Mullenbach et al. (2018) propose an attention based convolutional neural network (CNN) model to capture important information for each code. Xie and Xing (2018) adopt tree long short-term memory (LSTM) to utilize code descriptions. Though effective, they ignore the code hierarchy and code co-occurrence.

Hyperbolic Representation. Hyperbolic space has been applied to modeling complex networks (Krioukov et al., 2010). Recent research on representation learning demonstrates that the hyperbolic space is more suitable for representing symbolic data with hierarchical structures than the Euclidean space (Nickel and Kiela, 2017, 2018; Hamann, 2018). In the field of natural language processing (NLP), the hyperbolic representation has been successfully applied to question answering (Tay et al., 2018), machine translation (Gulcehre et al., 2018) and sentence representation (Dhingra et al., 2018). To our knowledge, this is the first work to apply hyperbolic representation method to the automatic ICD coding task.

Graph Convolutional Networks. GCN (Kipf and Welling, 2016) is a powerful neural network, which operates on graph data. It yields substantial improvements over various NLP tasks such as semantic role labeling (Marcheggiani and Titov, 2017), multi-document summarization (Yasunaga et al., 2017) and machine translation (Bastings et al., 2017). Veličković et al. (2017) propose graph attention networks (GAT) to summarize neighborhood features by using masked self-attentional layers. We are the first to capture the code co-occurrence characteristic via the GCN for the automatic ICD coding task.

3 Method

We propose a hyperbolic and co-graph representation (HyperCore) model for automatic ICD coding. Firstly, to capture the code hierarchy, we learn the code hyperbolic representations and measure the similarities between document and codes in the hyperbolic space. Secondly, to exploit code co-occurrence, we exploit the GCN to learn code co-occurrence representations and use them as query vectors to obtain code-aware document representations. Finally, the document-code similarity scores and code-aware document representations are then aggregated to predict the codes. Figure 3 shows the overall architecture of our proposed model.

3.1 Convolution Neural Network Encoder

We first map each word into a low dimensional word embedding space. The document can be denoted as $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, where N is the length of the document. Then, we exploit the CNN to encode the clinical text due to its high computational efficiency:

$$\boldsymbol{h}_i = \tanh(\boldsymbol{W}_c * \boldsymbol{x}_{i:i+k-1} + \boldsymbol{b}_c) \tag{1}$$

where $\boldsymbol{W}_c$ is the convolutional filter. $\boldsymbol{b}_c$ is the bias. k is the filter size. $*$ is the convolution operator.

3.2 Code-wise Attention

After encoding by CNN, we obtain the document representation $\boldsymbol{H} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \dots, \boldsymbol{h}_N\}$. Since we need to assign multiple codes for each document and different codes may focus on different sections of the document, we employ code-wise attention to learn relevant document representations for each code. We first generate the code vector for each code via averaging the word embeddings of its descriptor:

$$\boldsymbol{v}_i = \frac{1}{N_d} \sum\nolimits_{j=1}^{N_d} \boldsymbol{w}_j, \qquad i = 1, \dots, L \tag{2}$$

where $\boldsymbol{v}_i$ is the code vector, N_d is the length of the descriptor, $\boldsymbol{w}_j$ is the embedding of j-th word in the descriptor, and L is the total number of codes in the dataset (Jouhet et al., 2012; Johnson et al., 2016). The code vectors set is $\boldsymbol{V} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_L\}$.

Then, we generate the code-wise attention vector via matrix-vector product:

$$\boldsymbol{\alpha}_i = \mathrm{softmax}(\boldsymbol{H}^T \boldsymbol{v}_i) \tag{3}$$

Finally, we use the document representation $\boldsymbol{H}$ and attention vector $\boldsymbol{\alpha}_i$ to generate the code-aware document representation:

$$\boldsymbol{c}_i = \boldsymbol{H}\boldsymbol{\alpha}_i \tag{4}$$

We concatenate the $\boldsymbol{c}_i$ $(i = 1, \dots, L)$ to obtain the code-aware document representation, denoted as $\boldsymbol{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_L\} \in \mathbb{R}^{d_c \times L}$.

3.3 Document-Code Similarities in Hyperbolic Space

To capture the code hierarchy, we learn the code hyperbolic representations and measure the similarities between document and codes in the hyperbolic space. In this section, we propose a hyperbolic code embedder to obtain code hyperbolic representations, and we also propose a hyperbolic document projector to project the document representations from Euclidean space to hyperbolic space. We then compute the similarities between the document and codes in the hyperbolic space.

3.3.1 Hyperbolic Geometry

Hyperbolic geometry is a non-Euclidean geometry which studies spaces of constant negative curvature. Our approach is based on the Poincaré ball model (Nickel and Kiela, 2017), which is a particular model of hyperbolic space and is well-suited for gradient-based optimization. In particular, let $\mathcal{B}^n = \{\boldsymbol{x} \in \mathbb{R}^n \mid ||\boldsymbol{x}|| < 1\}$ be the open n-dimensional unit ball, where $|| \cdot ||$ denotes the Euclidean norm. The Poincaré ball $(\mathcal{B}^n, g_{\boldsymbol{x}})$ is defined by the Riemannian manifold, i.e., the open unit ball equipped with the Riemannian metric tensor:

$$g_{\boldsymbol{x}} = \left(\frac{2}{1 - ||\boldsymbol{x}||^2} \right)^2 g^E \tag{5}$$

where $\boldsymbol{x} \in \mathcal{B}^n$. g^E denotes the Euclidean metric tensor. Furthermore, the distance between two points $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{B}^n$ is given as:

$$d(\boldsymbol{u}, \boldsymbol{v}) = \mathrm{arcosh}(1 + 2\frac{||\boldsymbol{u} - \boldsymbol{v}||^2}{(1 - ||\boldsymbol{u}||^2)(1 - ||\boldsymbol{v}||^2)}) \tag{6}$$

where arcosh is the inverse hyperbolic cosine function, i.e., $\mathrm{arcosh}(x) = \ln(x + \sqrt{(x^2 - 1)})$. If we consider the origin $\boldsymbol{O}$ and two points $\boldsymbol{u}$, $\boldsymbol{v}$, when the two points moving towards the outside of the Poincaré ball (i.e., $||\boldsymbol{u}||, ||\boldsymbol{v}|| \to 1$), the distance $d(\boldsymbol{u}, \boldsymbol{v})$ tends to $d(\boldsymbol{u}, \boldsymbol{O}) + d(\boldsymbol{O}, \boldsymbol{v})$. That is, the path between the two points converges to a path through the origin, which can be seen as a tree-like hierarchical structure.

3.3.2 Hyperbolic Code Embedder

The tree-likeness of the hyperbolic space makes it natural to embed hierarchical structures. By embedding code hierarchy in the Poincaré ball, the top codes are placed near the origin and bottom codes are near the boundary. The embedding norm represents depth in the hierarchy, and the distance between embeddings represents the similarity. Let $\mathcal{D} = \{(l_p, l_q)\}$ be the set of parent-child relations between code pairs. $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_i\}_{i=1}^T, \boldsymbol{\theta}_i \in \mathcal{B}^{d_p}$ is the corresponding code embedding set, where T is the number of all ICD codes. In order to enforce related codes to be closer than unrelated codes, we minimize the following loss function to get the code hyperbolic representations when $||\boldsymbol{\theta}_i|| < 1 (i = 1, \dots, L)$:

$$\mathcal{J}(\boldsymbol{\Theta}) = - \sum_{(l_p, l_q) \in \mathcal{D}} \log \frac{\exp(-d(\boldsymbol{\theta}_p, \boldsymbol{\theta}_q))}{\sum_{l_{q'} \in \mathcal{N}(l_p)} \exp(-d(\boldsymbol{\theta}_p, \boldsymbol{\theta}_{q'}))} \tag{7}$$

where $\mathcal{N}(l_p) = \{l_{q'} | (l_p, l_{q'}) \notin \mathcal{D}\} \cup \{l_p\}$ is the set of negative samples. The hyperbolic code representations in our work are denoted as $\boldsymbol{\Theta}_L = \{\boldsymbol{\theta}_i\}_{i=1}^L$. $d(\cdot)$ is the distance defined as Equation (6).

3.3.3 Hyperbolic Document Projector

To compute the similarities between document and codes in hyperbolic space, the code-aware document representations $\boldsymbol{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_L\}$ need to be projected into the hyperbolic space. We exploit the re-parameterization technique (Dhingra et al., 2018; López et al., 2019) to implement it, which involves computing a direction vector $\boldsymbol{r}$ and a norm magnitude η. We use the $\boldsymbol{c}_i$ as an example to illustrate the procedure:

$$\begin{aligned} \overline{\boldsymbol{r}}_i &= \Phi_{dir}(\boldsymbol{c}_i), & \boldsymbol{r}_i &= \frac{\overline{\boldsymbol{r}}_i}{||\overline{\boldsymbol{r}}_i||} \\ \overline{\eta}_i &= \Phi_{norm}(\boldsymbol{c}_i), & \eta_i &= \sigma(\overline{\eta}_i) \end{aligned} \tag{8}$$

where $\Phi_{dir} : \mathbb{R}^{d_c} \rightarrow \mathbb{R}^{d_p}$ is the direction function. We parameterize it as a multi-layer perceptron (MLP). $\Phi_{norm} : \mathbb{R}^{d_c} \rightarrow \mathbb{R}$ is the norm magnitude function. We use a linear layer to implement it. σ is the sigmoid function to ensure the resulting norm $\eta_i \in (0, 1)$. The re-parameterized document representation is defined as $\boldsymbol{m}_i = \eta_i \boldsymbol{r}_i$, which lies in hyperbolic space $\mathcal{B}^{d_p}$.

The re-parameterization technique enables to project the code-aware document representation into the Poincaré ball, which enables the avoidance of the stochastic Riemannian optimization method (Bonnabel, 2013) to learn the parameters in the hyperbolic space. Instead, we can exploit the deep learning optimization method to update the parameters in the entire model.

3.3.4 Compute Document-Code Similarity

Since there doesn't exist a clear hyperbolic inner-product, the cosine similarity is not appropriate to be the metric. In our work, we adopt the hyperbolic distance function to model the relationships between the document and codes. Since the hyperbolic document representation for each code has been obtained, we just need to compute the similarity with the corresponding hyperbolic code embedding:

$$\begin{aligned} &score_i = d(\boldsymbol{m}_i, \boldsymbol{\theta}_i) \\ &\boldsymbol{S} = [score_1; score_2; \ldots; score_L] \end{aligned} \tag{9}$$

where $\boldsymbol{S} \in \mathbb{R}^L$ is the document-code similarity score. $[;]$ is the concatenation operation. $d(\cdot)$ is the distance function defined as Equation (6).

3.4 Code-aware Document Representations via Graph Convolutional Network

To exploit code co-occurrence, we exploit the graph to model code co-occurrence correlation, and then we use the GCN to learn code cooccurrence representations. In this section, we first construct the co-graph according to the statistics of the code co-occurrence in the training set, and then we exploit the GCN to encode the code co-occurrence correlation.

3.4.1 Code Co-graph Construction

Given a graph with L nodes, we can represent the graph using a $L \times L$ adjacency matrix $\boldsymbol{A}$. To capture the co-occurrence correlations between codes, we build the code co-occurrence graph (co-graph), which utilizes the code co-occurrence matrix as the adjacency matrix. If the i-th code and the j-th code co-occur in the clinical text, there is an edge between them. Intuitively, if the i-th code co-appears with the j-th code more often than the k-th code, the probabilities of the i-th code and the j-th code should have stronger dependencies. Therefore, in our work, we use the co-appearing times between two codes as the connection weights in the adjacency matrix, which can represent the prior knowledge. For example, if the i-th code co-appears n times with the j-th code, we set $\boldsymbol{A}_{ij} = n$.

3.4.2 Code Co-occurrence Encoding via GCN

The inputs of GCN are initial representations of codes $\boldsymbol{V}$ which are obtained via Equation (2) and the adjacency matrix $\boldsymbol{A}$. We use the standard convolution computation (Kipf and Welling, 2016) to encode code co-occurrence:

$$\boldsymbol{H}^{(l+1)} = \rho(\tilde{\boldsymbol{D}}^{-\frac{1}{2}} \tilde{\boldsymbol{A}} \tilde{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{H}^{(l)} \boldsymbol{W}^{(l)}) \tag{10}$$

where $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$. $\boldsymbol{I}$ is the identity matrix, $\tilde{\boldsymbol{D}}_{ii} = \sum_j \tilde{\boldsymbol{A}}_{ij}$, $\boldsymbol{H}^{(l)} \in \mathbb{R}^{L \times d_c}$ and $\boldsymbol{H}^{(0)} = \boldsymbol{V}$. ρ is an activation function (e.g., ReLU). After co-occurrence correlation encoding via GCN, the code representations enable to capture the code co-occurrence correlations. Then, we use the code-wise attention to obtain code-aware document representations, denoted as $\boldsymbol{D} = \{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_L\}$[1].

3.5 Aggregation Layer

After capturing the code hierarchy and code co-occurrence, we use an aggregation layer to fuse document-code similarity scores $\boldsymbol{S}$ and code-aware document representations $\boldsymbol{D}$ for enhancing representation with each other:

$$\boldsymbol{U} = \lambda \boldsymbol{W}_s \boldsymbol{S} + \boldsymbol{D}^T \boldsymbol{W}_d \tag{11}$$

where $\boldsymbol{W}_s$ and $\boldsymbol{W}_d$ are transformation matrixes. $\boldsymbol{U} = \{u_1, u_2, \ldots, u_L\} \in \mathbb{R}^L$ are final document representations for each code. λ is the hyperparameter.

[1] C and D are both code-aware document representations, but D captures the code co-occurrence correlations.

Model	MIMIC-III full						MIMIC-III 50				
	AUC		F1		P@N		AUC		F1		P@5
	Macro	Micro	Macro	Micro	8	15	Macro	Micro	Macro	Micro	
C-MemNN	–	–	–	–	–	–	0.833	–	–	–	0.420
C-LSTM-ATT	–	–	–	–	–	–	–	0.900	–	0.532	–
CAML	0.895	0.986	0.088	0.539	0.709	0.561	0.875	0.909	0.532	0.614	0.609
DR-CAML	0.897	0.985	0.086	0.529	0.690	0.548	0.884	0.916	0.576	0.633	0.618
HyperCore	**0.930** ±0.001	**0.989** ±0.005	**0.090** ±0.003	**0.551** ±0.001	**0.722** ±0.002	**0.579** ±0.001	**0.895** ±0.003	**0.929** ±0.002	**0.609** ±0.001	**0.663** ±0.001	**0.632** ±0.002

Table 1: Comparison of our model and other baselines on the MIMIC-III dataset. We run our model 10 times and each time we use different random seeds for initialization. We report the *mean* ± *standard deviation* of each result.

3.6 Training

The prediction for each code is generated via:

$$\hat{y_i} = \sigma(u_i), \qquad i = 1, \ldots, L \qquad (12)$$

Our model is to be trained using a multi-label binary cross-entropy loss:

$$\mathcal{L} = \sum\nolimits_{i=1}^{L} [-y_i \log(\hat{y_i}) - (1 - y_i)\log(1 - \hat{y_i})] \qquad (13)$$

where $y_i \in \{0, 1\}$ is the ground truth for the i-th code.

4 Experiments

4.1 Datasets

We evaluate our proposed model on two widely used datasets, including MIMIC-II (Jouhet et al., 2012) and MIMIC-III (Johnson et al., 2016). Both datasets contain discharge summaries that are tagged by human coders with a set of ICD-9 codes. For MIMIC-III dataset, we use the same experimental setting as previous works (Shi et al., 2017; Mullenbach et al., 2018). The dataset has two common settings: MIMIC-III full and MIMIC-III 50. For MIMIC-III full setting, the setting consists of 8921 codes, 47719, 1631 and 3372 discharge summaries for training, development and testing respectively. For MIMIC-III 50 setting, the setting contains the top 50 most frequent codes, 8067, 1574 and 1730 discharge summaries for training, development and testing respectively. For the MIMIC-II dataset, we use the same splits as previous works (Perotte et al., 2013; Mullenbach et al., 2018), there are 20533 and 2282 clinical notes for training and testing, and 5031 unique ICD-9 codes in the dataset.

4.2 Metrics and Parameter Settings

Following previous work (Mullenbach et al., 2018), we use macro-averaged and micro-averaged F1, macro-averaged and micro-averaged AUC (area under the ROC, i.e., receiver operating characteristic curve) and Precision@N (P@N) as the metrics.

Model	AUC		F1		P@8
	Macro	Micro	Macro	Micro	
SVM	–	–	–	0.293	–
HA-GRU	–	–	–	0.366	–
CAML	0.820	0.966	0.048	0.442	0.523
DR-CAML	0.826	0.966	0.049	0.457	0.515
HyperCore	**0.885** ±0.001	**0.971** ±0.004	**0.070** ±0.002	**0.477** ±0.003	**0.537** ±0.003

Table 2: Experimental results are shown in *means* ± *standard deviations* on the MIMIC-II dataset.

The P@N indicates the proportion of the correctly-predicted labels in the top-N predicted labels.

Hyper-parameters are tuned on the development set by grid search. The word embedding size d_e is 100. The convolution filter size is 10. The size of the filter output is 200. The dropout rate is 0.4. The λ is 0.2. The batch size is 16. Adam (Kingma and Ba, 2014) is used for optimization with an initial learning rate 1e-4. We pre-train the word embeddings on the combination of training sets of MIMIC-II and MIMIC-III datasets by using word2vec toolkit (Mikolov et al., 2013).

4.3 Baselines

SVM: A hierarchical support vector machine (SVM) is proposed by Perotte et al. (2013) to use the hierarchical nature of ICD codes, which is evaluated on the MIMIC-II dataset.

C-MemNN: A condensed memory neural network is proposed by Prakash et al. (2017) to predict ICD codes on the MIMIC-III 50 dataset.

C-LSTM-ATT: A character-aware LSTM based attention model is proposed by Shi et al. (2017). It is also evaluated on the MIMIC-III 50 dataset.

HA-GRU: A hierarchical attention gated recurrent unit model is proposed by Baumel et al. (2018) to predict ICD codes on the MIMIC-II dataset.

CAML & DR-CAML: The **c**onvolutional **a**ttention network for **m**ulti-**l**abel classification (CAML) is proposed by Mullenbach et al. (2018). DR-CAML is an extension of CAML which

Models	MIMIC-III full		MIMIC-III 50		MIMIC-II	
	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1
HyperCore	**0.090**	**0.551**	**0.609**	**0.663**	**0.070**	**0.477**
w/o hyperbolic representation	0.081	0.539	0.576	0.645	0.062	0.464
w/o co-graph representation	0.085	0.541	0.582	0.637	0.055	0.453
w/o hyperbolic and co-graph representation	0.077	0.531	0.570	0.626	0.047	0.439

Table 3: Ablation study by removing the main components, where “w/o” indicates without.

incorporates the code description. They achieve the state-of-the-art performance on the MIMIC-III and MIMIC-II datasets.

4.4 Compared with State-of-the-art Methods

We repeat 10 times training and each time we use different random seeds for initialization. We report the *mean* ± *standard deviation* of each result. Table 1 and Table 2 show the results on the MIMIC-III and MIMIC-II datasets, respectively. Since some baselines are evaluated either on MIMIC-III or MIMIC-II, the baselines used for the two datasets are different. Overall, we observe that:

(1) In Table 1, our method HyperCore outperforms all the baselines on MIMIC-III dataset. For example, compared with the state-of-the-art model DR-CAML, our method achieves 2.2% and 3% improvements of Micro-F1 score on MIMIC-III full and MIMIC-III 50 respectively. It indicates that, as compared to neural network based models that handle each code in isolation, our method can better take advantage of the rich correlations among codes. In addition, the small standard deviations indicate that our model obtains stable good results.

(2) As previous work (Mullenbach et al., 2018), the Macro-F1 score of our method on MIMIC-III full is lower than that on the MIMIC-III 50. The reason is that MIMIC-III full has long-tail frequency distributions, and the Macro-F1 places more emphasis on rare code prediction. Therefore, it is difficult to achieve a high Macro-F1 score on MIMIC-III full. Nevertheless, our method still achieves the best result on the Macro-F1 metric. It indicates that our method is very effective.

(3) In Table 2, our method HyperCore also achieves the best performance over all metrics on the MIMIC-II. Especially, compared with the state-of-the-art model DR-CAML, our method achieves 5.9% improvements of Macro-AUC, which indicates the effectiveness of our method.

(4) As shown in Table 2, the neural network based methods outperform the traditional model (SVM), which indicates the limitation of human-designed features and the advancement of neural networks for the automatic ICD coding.

4.5 Ablation Experiment

To investigate the effectiveness of the hyperbolic and co-graph representation, we conduct the ablation studies. The experimental results are listed in Table 3. From the results, we can observe that:

(1) **Effectiveness of Hyperbolic Representation.** Compared with the model removed hyperbolic representation, the HyperCore improves the Micro-F1 score from 0.539 to 0.551 on MIMIC-III full dataset. It demonstrates the effectiveness of the hyperbolic representation.

(2) **Effectiveness of Co-graph Representation.** Compared with the model removed the co-graph representation, the HyperCore model improves the performance, achieving 2.6% improvements of Micro-F1 score on the MIMIC-III 50 dataset. The great improvements indicate the co-graph representation is very effective.

(3) **Effectiveness of Hyperbolic and Co-graph Representation.** When we remove the hyperbolic and co-graph representation, the performance drops significantly. The Micro-F1 score drops from 0.477 to 0.439 on the MIMIC-II dataset. It indicates that simultaneously exploiting the hyperbolic and co-graph representation is also very effective.

4.6 Discussion

4.6.1 The Analysis of Hyperbolic Code Embedding Dimension

Since the dimensionality of the hperbolic code embeddings is very important for hyperbolic representation, we investigate its effect. The size of hyperbolic code embeddings is set 10, 20, 50, 70 and 100. Table 4 shows the results of our model on the MIMIC-III and MIMIC-II datasets. We have two important observations:

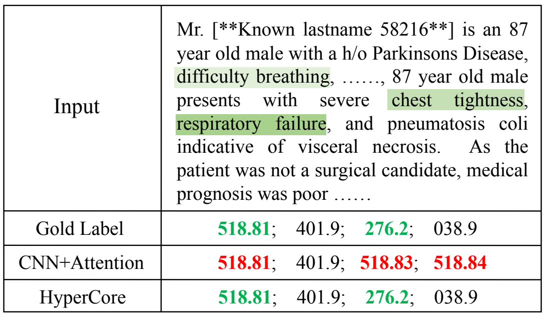
(1) The best hyperbolic code embedding dimensionality on MIMIC-III full is larger than it on MIMIC-III 50 and MIMIC-II. The reason may be that the number of codes in MIMIC-III full is

Dimensionality	MIMIC-III full			MIMIC-III 50			MIMIC-II		
	Macro-F1	Micro-F1	P@8	Macro-F1	Micro-F1	P@5	Macro-F1	Micro-F1	P@8
10	0.083	0.539	0.701	0.593	0.651	0.619	0.064	0.463	0.528
20	0.085	0.542	0.704	0.598	0.656	0.625	0.066	0.471	0.532
50	0.087	0.547	0.708	**0.609**	**0.663**	**0.632**	**0.070**	**0.477**	**0.537**
70	**0.090**	**0.551**	**0.722**	0.605	0.660	0.627	0.065	0.473	0.534
100	0.083	0.548	0.710	0.602	0.659	0.625	0.064	0.473	0.530

Table 4: Experimental results of HyperCore with different size of hyperbolic code embeddings.

ICD-9 code	Norm
460-519 (*Diseases of the Respiratory System*)	0.455
480-488 (*Pneumonia and Influenza*)	0.520
487 (*Influenza*)	0.568
487.8 (*Influenza with other manifestations*)	0.928
520-579 (*Diseases of the Digestive System*)	0.412
550-579 (*Hernia of Abdominal Cavity*)	0.472
550 (*Inguinal hernia*)	0.590
550.0 (*Inguinal hernia with gangrene*)	0.902

Table 5: The first and second blocks list some codes and their hyperbolic norms of '*'Diseases of the Respiratory System*" and "*Diseases of the Digestive System*", respectively. In each block, the disease becomes more specific from top to bottom. The norms of codes increase with the depth.



Input	Mr. [**Known lastname 58216**] is an 87 year old male with a h/o Parkinsons Disease, difficulty breathing, ……, 87 year old male presents with severe chest tightness, respiratory failure, and pneumatosis coli indicative of visceral necrosis. As the patient was not a surgical candidate, medical prognosis was poor ……
Gold Label	**518.81**; 401.9; **276.2**; 038.9
CNN+Attention	**518.81**; 401.9; **518.83**; **518.84**
HyperCore	**518.81**; 401.9; **276.2**; 038.9

Figure 4: An example to illustrate the effectiveness of the proposed model. The green bold codes indicate they are highly correlated. The red bold codes denote there exists contradictions between them.

more than other two datasets, which needs higher-dimensional hyperbolic code embedding to represent the code hierarchy.

(2) The performance does not always improve when the hyperbolic code embedding size increases. We guess that low dimensional embeddings can capture the hierarchy and the network is prone to over-fitting when high dimensional hyperbolic code embeddings are used.

4.6.2 The Hierarchy of Hyperbolic Code Embedding

After embedding the ICD codes into the hyperbolic space, the top level codes will be placed near the origin and low level codes near the boundary, which can be reflected via their norms. Table 5 shows examples of ICD-9 codes and their hyperbolic norms. The first and second blocks list codes of "*Diseases of the Respiratory System*" and "*Diseases of the Digestive System*", respectively. As expected, the lower level codes have higher hyperbolic norms, and this approves that when the disease is more specific, the hyperbolic norm is larger. For example, code "487.8 (*influenza with other manifestations*)" has a higher norm than "487 (*influenza*)", and "550.0 (*inguinal hernia with gangrene*)" has a higher norm than "550 (*inguinal hernia*)". It indicates that the hyperbolic code embeddings can capture the code hierarchy.

4.7 Case Study

We give an example shown in Figure 4 to illustrate the visualization of code-wise attention and the effectiveness of hyperbolic and co-graph representation. (1) **Code-wise attention visualization**: When the HyperCore model predicts the code "518.81 (*acute respiratory failure*)", it can assign larger weights to more informative words, like "respiratory failure" and "chest tightness". It shows the codes-wise attention enables to select the most informative words. (2) **The effectiveness of hyperbolic representations**: Our proposed model and the CNN+Attention can both correctly predict the code "518.81". However, the CNN+Attention model gives contradictory predictions. Our proposed model can avoid the prediction contradictions by exploiting code hierarchy, which proves the effectiveness of hyperbolic representations. (3) **The effectiveness of co-graph representation**: Although there is no very obvious clue to predict the code "276.2 (*acidosis*)", our model can exploit the co-occurrence between the code "518.81" and "276.2" to assist in inferring the code "276.2". It demonstrates the effectiveness of the co-graph representation.

5 Conclusion

In this paper, we propose a novel hyperbolic and co-graph representation framework for the automatic ICD coding task, which can jointly exploit code hierarchy and code co-occurrence. We exploit the hyperbolic representation learning method to leverage the code hierarchy in the hyperbolic space. Moreover, we use the graph convolutional network to capture the co-occurrence correlation. Experimental results on two widely used datasets indicate that our proposed model outperforms previous state-of-the-art methods. We believe our method can also be applied to other tasks that need to exploit hierarchical label structure and label co-occurrence, such as fine-grained entity typing and hierarchical multi-label classification.

Acknowledgments

This work is supported by the Natural Key R&D Program of China (No.2017YFB1002101), the National Natural Science Foundation of China (No.61922085, No.61533018, No.61976211, No.61806201) and the Key Research Program of the Chinese Academy of Sciences (Grant NO. ZDBS-SSW-JSC006). This work is also supported by Beijing Academy of Artificial Intelligence (BAAI2019QN0301) and the CCF-Tencent Open Research Fund.

References

Anand Avati, Kenneth Jung, Stephanie Harman, Lance Downing, Andrew Ng, and Nigam H Shah. 2018. Improving palliative care with deep learning. *BMC medical informatics and decision making*, 18(4):122.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967.

Tal Baumel, Jumana Nassour-Kassis, Raphael Cohen, Michael Elhadad, and Noemie Elhadad. 2018. Multi-label classification of patient notes: case study on icd code assignment. In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*.

Silvere Bonnabel. 2013. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229.

Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*, pages 301–318.

Bhuwan Dhingra, Christopher Shallue, Mohammad Norouzi, Andrew Dai, and George Dahl. 2018. Embedding text in hyperbolic spaces. In *Proceedings of the Twelfth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-12)*, pages 59–69.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, et al. 2018. Hyperbolic attention networks. In *Proceedings of International Conference on Learning Representations*.

Matthias Hamann. 2018. On the tree-likeness of hyperbolic spaces. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 164, pages 345–361. Cambridge University Press.

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*, 3:160035.

Vianney Jouhet, Georges Defossez, Anita Burgun, Pierre Le Beux, P Levillain, Pierre Ingrand, and Vincent Claveau. 2012. Automated classification of free-text pathology reports for registration of incident cases of cancer. *Methods of information in medicine*, 51(03):242–251.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. In *Proceedings of International Conference on Learning Representations*.

Bevan Koopman, Guido Zuccon, Anthony Nguyen, Anton Bergheim, and Narelle Grayson. 2015. Automatic icd-10 classification of cancers from free-text death certificates. *International journal of medical informatics*, 84(11):956–965.

Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. 2010. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106.

Dee Lang. 2007. Consultant report-natural language processing in the health care industry. *Cincinnati Children's Hospital Medical Center, Winter*, 6.

Leah S Larkey and W Bruce Croft. 1996. Combining classifiers in text categorization. In *SIGIR*, pages 289–297. Citeseer.

Federico López, Benjamin Heinzerling, and Michael Strube. 2019. Fine-grained entity typing in hyperbolic space. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 169–180. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

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.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347.

Maximillian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, pages 3776–3785.

Adler Perotte, Rimma Pivovarov, Karthik Natarajan, Nicole Weiskopf, Frank Wood, and Noémie Elhadad. 2013. Diagnosis code assignment: models and evaluation metrics. *Journal of the American Medical Informatics Association*, 21(2):231–237.

Aaditya Prakash, Siyuan Zhao, Sadid A Hasan, Vivek Datla, Kathy Lee, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2017. Condensed memory networks for clinical diagnostic inferencing. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3274–3280. AAAI Press.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3132–3142.

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*.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 583–591. ACM.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. In *Proceedings of International Conference on Learning Representations*.

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.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462.

Ying Yu, Min Li, Liangliang Liu, Zhihui Fei, Fang-Xiang Wu, and Jianxin Wang. 2019. Automatic icd code assignment of chinese clinical notes based on multilayer attention birnn. *Journal of biomedical informatics*, 91:103114.