Hyperbolic Interaction Model for Hierarchical Multi-Label Classification

Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, Liping Jing
Beijing Jiaotong University, Beijing, China
{18120345, 18120367, 17112079, 18120340, ljjing}@bjtu.edu.cn

Abstract
Different from the traditional classification tasks which assume mutual exclusion of labels, hierarchical multi-label classification (HMLC) aims to assign multiple labels to every instance with the labels organized under hierarchical relations. Besides, the labels, since linguistic ontologies are intrinsically hierarchies, the conceptual relations between words can also form hierarchical structures. Thus it can be a challenge to learn mappings from word hierarchies to label hierarchies. We propose to model the word and label hierarchies by embedding them jointly in the hyperbolic space. The main reason is that the tree-likeness of the hyperbolic space matches the complexity of symbolic data with hierarchical structures. A new Hyperbolic Interaction Model (HyperIM) is designed to learn the label-aware document representations and make predictions for HMLC. Extensive experiments are conducted on three benchmark datasets. The results have demonstrated that the new model can realistically capture the complex data structures and further improve the performance for HMLC comparing with the state-of-the-art methods. To facilitate future research, our code is publicly available.

Introduction
Traditional classification methods suppose the labels are mutually exclusive, whereas for hierarchical classification, labels are not disjointed but organized under a hierarchical structure. Such structure can be a tree or a Directed Acyclic Graph, which indicates the parent-child relations between labels. Typical hierarchical classification tasks include protein function prediction in bioinformatics tasks (Wehmann et al. 2017), image annotation (Dimitrovski et al. 2011) and text classification (Meng et al. 2019). In this paper, we focus on hierarchical multi-label text classification, which aims to assign multiple labels to every document instance with the labels hierarchically structured.

In multi-label classification (MLC), there usually exist a lot of infrequently occurring tail labels (Bhatia et al. 2015), especially when the label sets are large. The fact that tail labels lack of training instances makes it hard to train an efficacious classifier. Fortunately, the effectiveness of utilizing label correlations to address this problem has lately been demonstrated. In literatures, label correlations can be determined from label matrix (Zhang et al. 2018) or label content (Wang et al. 2018). The main idea is to project the labels into a latent vectorial space, where each label is represented as a dense low-dimensional vector, so that the label correlations can be characterized in this latent space. For hierarchical multi-label classification (HMLC), labels are organized into a hierarchy and located at different hierarchical levels accordingly. Since a parent label generally has several child labels, the number of labels grows exponentially in child levels. In some special cases, most labels are located at the lower levels, and few training instances belong to each of them. In other words, tail labels also exist in HMLC. Different from the traditional MLC, the label structure, which is intuitively useful to detect label correlations, is well provided in HMLC.

Inspired by recent works on learning hierarchical representations (Nickel and Kiela 2017), we propose to embed the label hierarchy in the hyperbolic space. Taking advantage of the hyperbolic representation capability, we design a Hyperbolic Interaction Model (HyperIM) to classify hierarchically structured labels. HyperIM embeds both document words and labels jointly in the hyperbolic space to preserve their latent structures (e.g. structures of conceptual relations between words and parent-child relations between labels). Semantic connections between words and labels can be furthermore explicitly measured according to the word and label embeddings, which benefits extracting the most related components from documents and constructing the label-aware document representations. The prediction is directly optimized by minimizing the cross-entropy loss. Our contributions are summarized as follows:

• We adopt hyperbolic space to improve HMLC. A novel model HyperIM is designed to embed the label hierarchy and the document text in the same hyperbolic space. For the classification, semantic connections between words and labels are explicitly measured to construct the label-aware document representations.
• We present partial interaction to improve the scalability of the interaction model. For large label spaces, negative sampling is used to reduce the memory usage during interaction.
Extensive experiments on three benchmark datasets show the effectiveness of HyperIM. An ablation test is performed to demonstrate the superiority of the hyperbolic space over the Euclidean space for HMLC. In addition, our code is publicly available.

Preliminaries

Let $X$ denote the document instance space, and let $L = \{l_i\}_{i=1}^C$ denote the finite set of $C$ labels. Labels are organized under a hierarchical structure in HMLC, $T = \{(l_p, l_q) \mid l_p \geq l_q\}$ denotes their parent-child relations, where $l_p$ is the parent of $l_q$. Given the text sequence of a document instance $x \in X$ and its one-hot ground truth label vector $y \in \{0, 1\}^C$, the classification model learns the document-label similarities, i.e., the probabilities for all the labels given the document. Let $p \in [0, 1]^C$ denote the label probability vector predicted by the model for $x$, where $p[i]$ is used to denote the $i$-th element in a vector. The model can be trained by optimizing certain loss function that compares $y$ and $p$.

To capture the fine-grained semantic connections between a document instance and the labels, the document-label similarities are obtained by aggregating the word-label similarities. More specifically, for the text sequence with $T$ word tokens, i.e., $x = [x_1, \ldots, x_T]$, the $i$-th label-aware document representation $s_i = [\text{score}(x_1, l_i); \ldots; \text{score}(x_T, l_i)]$ can be calculated via certain score function. $p[i]$ is then deduced from $s_i$. This process is adapted from the interaction mechanism (Du et al. 2019), which is usually used in tasks like natural language inference (Wang and Jiang 2016). Based on the idea that labels can be considered as abstraction from their word descriptions, sometimes a label is even a word itself, the word-label similarities can be derived from their embeddings in the latent space by the same way as the word similarity, which is widely studied in word embedding methods such as GloVe (Pennington, Socher, and Manning 2013).

Note that word embeddings are insufficient to fully represent the meanings of words, especially in the case of word-sense disambiguation (Navigli 2009). Take the word “bank” as an example, it has significantly different meanings in the text sequences ”go to the bank and change some money” and ”flowers generally grow on the river bank”, which will cause a variance when matching with labels ”economy” and ”environment”. In order to capture the real semantics of each word, we introduce RNN-based word encoder which can take the contextual information of text sequences into consideration.

The Poincar Ball

In HyperIM, both document text and labels are embedded in the hyperbolic space. The hyperbolic space is a homogeneous space that has a constant negative sectional curvature, while the Euclidean space has zero curvature. The hyperbolic space can be described via Riemannian geometry (Hopper and Andrews 2011). Following previous works (Nickel and Kiela 2017) (Ganea, Becigneul, and Hofmann 2018) (Tifrea, Bécigneul, and Ganea 2019), we adopt the Poincaré ball.

An $n$-dimensional Poincaré ball $(B^n, g^B)$ is a subset of $\mathbb{R}^n$ defined by the Riemannian manifold $B^n = \{x \in \mathbb{R}^n \mid \|x\| < 1\}$ equipped with the Riemannian metric $g^B$, where $\|\cdot\|$ denotes the Euclidean $L^2$ norm. As the Poincaré ball is conformal to the Euclidean space (Cannon et al. 1997), the Riemannian metric can be written as $g^B = \lambda^2 g^E$ with the conformal factor $\lambda := \frac{2}{1-\|p\|^2}$ for all $p \in B^n$, where $g^E = I_n$ is the Euclidean metric tensor. It is known that the geodesic distance between two points $u, v \in B^n$ can be induced using the ambient Euclidean geometry as $d_{g^E}(u, v) = \cosh^{-1}(1 + \frac{1}{2} \lambda u \cdot v - \|u - v\|^2)$. This formula demonstrates that the distance changes smoothly w.r.t. $\|u\|$ and $\|v\|$, which is key to learn continuous embeddings for hierarchical structures.

With the purpose of generalizing operations for neural networks in the Poincaré ball, the formalism of the Möbius gyrovector space is used (Ganea, Becigneul, and Hofmann 2018). The Möbius addition for $u, v \in B^n$ is defined as $u \oplus v = \frac{(1+2(u,v) + \|v\|^2)u + (1-\|u\|^2)v}{1+2(u,v) + \|u\|^2\|v\|^2}$, where $(\cdot, \cdot)$ denotes the Euclidean inner product. The Möbius addition operation in the Poincaré disk $B^2$ (2-dimensional Poincaré ball) can be visualized in Figure 1a. Then the Poincaré distance can be visualized in Figure 1b.
rewritten as
\[ d_{B^n}(u, v) = 2 \tanh^{-1}(|u + v|). \] (1)

The Möbius matrix-vector multiplication for \( M \in \mathbb{R}^{m \times n} \) and \( p \in B^n \) when \( Mp \neq 0 \) is defined as \( M \otimes p = \text{tanh}(\frac{\|Mp\|}{\|p\|} \tanh^{-1}(\|p\|)) \frac{Mp}{\|Mp\|} \), and \( M \otimes p = 0 \) when \( Mp = 0 \). Moreover, the closed-form derivations of the exponential map \( \exp_p : T_pB^n \rightarrow B^n \) and the logarithmic map \( \log_p : B^n \rightarrow T_pB^n \) for \( p \in B^n \), \( w \in T_pB^n \setminus \{0\} \), \( u \in B^n \setminus \{p\} \) are given as \( \exp_p(w) = p + \left( \sum_{n} \left( \frac{\|w\|}{\|p\|} \tanh\left( \frac{\|w\|}{\|p\|} \right) \right)^{n} \right) u \) and \( \log_p(u) = \frac{1}{2p} \tanh^{-1}(\|p + u\|) - \frac{p^n u}{\|p^n u\|} \).

These operations make hyperbolic neural networks available (Ganea, Bécigneul, and Hofmann 2018) and gradient-based optimizations can be performed to estimate the model parameters in the Poincar ball (Bécigneul and Ganea 2019).

**Hyperbolic Interaction Model**

We design a Hyperbolic Interaction Model (HyperIM) for hierarchical multi-label text classification. Given the text sequence of a document, HyperIM measures the word-label similarities by calculating the geodesic distance between the jointly embedded words and labels in the Poincar ball. The word-label similarity scores are then aggregated to estimate the label-aware document representations and further predict the probability for each label. Figure 2 demonstrates the framework of HyperIM.

**Hyperbolic Label Embedding**

The tree-likeness of the hyperbolic space (Hamann 2018) makes it natural to embed hierarchical structures. For instance, Figure 16 presents a tree embedded in the Poincar disk, where the root is placed at the origin and the leaves are close to the boundary. It has been shown that any finite tree can be embedded with arbitrary low distortion into the Poincar ball while the distances are approximately preserved (Sarkar 2011). Conversely, it’s difficult to perform such embedding in the Euclidean space even with unbounded dimensionality (Sala et al. 2018). Since the label hierarchy is defined in the set \( T = \{l_p, l_q \mid l_p \geq l_q, l_p, l_q \in \mathcal{L}\} \), the goal is to maximize the distance between labels without parent-child relation (Nickel and Kiela 2017). Let \( \Theta^L = \{\theta^l_i\}_{i=1}^C \), \( \theta^l \in B^n \) be the label embedding set, using Riemannian adaptive optimization methods (Bécigneul and Ganea 2019), \( \Theta^L \) can be efficiently estimated by minimizing the loss function

\[ \mathcal{L}^h_{loss} = - \sum_{(l_p, l_q) \in \mathcal{T}} \log \frac{\exp(-d_{B^n}(\theta^l_i, \theta^l_j))}{\sum_{l_{q'} \in \mathcal{N}(l_p)} \exp(-d_{B^n}(\theta^l_i, \theta^l_{q'}))}, \] (2)

where \( \mathcal{N}(l_p) = \{l_{q'} \mid (l_p, l_{q'}) \notin \mathcal{T} \} \cup \{l_p\} \) is the set of negative samples. The obtained \( \Theta^L \) can capture the hierarchical structure among labels.

**Hyperbolic Word Embedding**

For natural language processing, word embeddings are essential in neural networks as intermediate features. Given the statistics of word co-occurrences in the corpus, we adopt the Poincar GloVe (Tifrea, Bécigneul, and Ganea 2019) to capture the elementary relations between words by embedding them in the hyperbolic space. Let \( X_i \) indicate the times that word \( i \) and word \( j \) co-occur in the same context window, \( \theta^w_i \in B^n \) be the target embedding vector in the \( k \)-dimensional Poincar ball for word \( i \), and \( \theta^w_j \in B^n \) be the context embedding vector for word \( j \). With the aid of Riemannian adaptive optimization methods, the embeddings \( \Theta^w = \{\theta^w_i\}_{i=1}^V \) and \( \Theta^E = \{\theta^E_j\}_{j=1}^V \) for the corpus with vocabulary size \( V \) are estimated by minimizing the loss function

\[ \mathcal{L}^w_{loss} = \sum_{i,j=1}^{V} f(X_{ij})(-h(d_{B^n}(\theta^w_i, \theta^w_j)) + b_i + b_j - \log(X_{ij}))^2, \] (3)

where \( b_i, b_j \) are the biases, and the two suggested weight functions are defined as \( f(x) = \min(1, (x/100)^3/4) \), \( h(x) = \cosh^2(x) \).

Since the WordNet hypernym (Miller 1995) set \( \mathcal{T}^w = \{x_p, x_q \mid x_p \geq x_q\} \), where word \( x_p \) is the hypernym of word \( x_q \) in the corpus, is similar to the label hierarchy, providing the hypernym information to the word embeddings, latent correlations between the two hierarchies can be later captured via interaction. Considering the learned Poincar GloVe embeddings \( \Theta^E \) don’t explicitly capture the conceptual relations among words, a post-processing step similar to Eq. (2) on top of \( \Theta^E \) is further conducted by using Riemannian adaptive optimization methods to minimize the loss function

\[ \mathcal{L}^e_{loss} = - \sum_{(x_p, x_q) \in \mathcal{T}} \log \frac{\exp(\sum_{x_{q'} \in \mathcal{N}(x_p)} \exp(-d_{B^n}(\theta^w_{q'}, \theta^w_{q''})))}{\sum_{x_{q'} \in \mathcal{N}(x_p)} \exp(\sum_{x_{q''} \in \mathcal{N}(x_p)} \exp(-d_{B^n}(\theta^w_{q'}, \theta^w_{q''})))}, \] (4)

where \( \mathcal{N}(x_p) = \{x_{q'} \mid (x_p, x_{q''}) \notin \mathcal{T}^w \} \cup \{x_p\} \) is the set of negative samples.

**Hyperbolic Word Encoder**

Considering the word-sequence disambiguation (Navigli 2009), meanings of polysemous words are difficult to distinguish if the word and label embeddings interact with each other directly, since all the meanings of a word are embedded on the same position. However, polysemous words can usually be inferred from the context.

Given the text sequence of a document with \( T \) word tokens \( x = \{x_1, \ldots, x_T\} \), pre-trained hyperbolic word embeddings \( \Theta^E \) can be used to learn the final word representations according to the text sequence. To consider the sequentiality of the text sequence, we take advantage of the hyperbolic space adaptive RNN-based architectures (Ganea, Bécigneul, and Hofmann 2018). More specifically, given \( \Theta^e = \{\theta^e_{1}, \ldots, \theta^e_{T}\} \) where \( \theta^e_i \in \Theta^E (t = 1, \ldots, T) \), the hyperbolic word encoder based on the GRU architecture adjusts the embedding for each word to fit its context via

\[
\begin{align*}
    r_t &= \sigma(\log_{0}(W^{r} \otimes \theta^e_{t-1} \oplus U^{r} \otimes \theta^r_t \oplus b^{r})), \\
    z_t &= \sigma(\log_{0}(W^{z} \otimes \theta^e_{t-1} \oplus U^{z} \otimes \theta^z_t \oplus b^{z})), \\
    \theta^e_t &= \varphi((W^{g} \text{diag}(r_t)) \otimes \theta^e_{t-1} \oplus U^{g} \otimes \theta^g_t \oplus b^{g}), \\
    \theta^r_t &= \theta^r_{t-1} \oplus \text{diag}(z_t) \otimes (\theta^r_{t-1} \oplus \theta^r_t),
\end{align*}
\] (5)
where \( \Theta^w = [\theta^w_1, \ldots, \theta^w_T] \) denotes the encoded embeddings for the text sequence, the initial hidden state \( \theta^w_0 := 0 \), \( r_t \) is the reset gate, \( z_t \) is the update gate, \( \text{diag}(\cdot) \) denotes the diagonal matrix with each element in the vector on its diagonal, \( \sigma \) is the sigmoid function, \( \varphi \) is a pointwise non-linearity, typically \( \text{sigmoid}, \text{tanh} \) or \( \text{ReLU} \). Since the hyperbolic space naturally has non-linearity, \( \varphi \) can be identity (no non-linearity) here.

The formula of the hyperbolic GRU is derived by connecting the Möbius gyrovector space with the Poincar ball (Ganea, Becigneul, and Hofmann 2018). The six weights \( W, U \in \mathbb{R}^{k \times k} \) are trainable parameters in the Euclidean space and the three biases \( b \in \mathbb{R}^{k} \) are trainable parameters in the hyperbolic space (the superscripts are omitted for simplicity). Thus the weights \( W \) and \( U \) are updated via vanilla optimization methods, and the biases \( b \) are updated with Riemannian adaptive optimization methods. \( \Theta^w \) will be used for measuring the word-label similarities during the following interaction process.

**Interaction in the Hyperbolic Space**

The major objective of text classification is to build connections from the word space to the label space. In order to capture the fine-grained semantic information, we first construct the label-aware document representations, and then learn the mappings between the document instance and the labels.

**Label-aware Document Representations** Once the encoded word embeddings \( \Theta^w \) and label embeddings \( \Theta^l \) are obtained, it’s expected that every pair of word and label embedded close to each other based on their geodesic distance if they are semantically similar. Note that cosine similarity (Wang, Hamza, and Florian 2017) is not appropriate to be the metric since there doesn’t exist a clear hyperbolic inner-product for the the Poincaré ball (Tifrea, Becigneul, and Ganea 2019), so the geodesic distance is more intuitively suitable. The similarity between the \( t \)-th word \( x_t(t = 1, \ldots, T) \) and the \( i \)-th label \( l_i(i = 1, \ldots, C) \) is calculated as:

\[
\text{score}(x_t, l_i) = -d_{gh}(\theta^w_t, \theta^l_i),
\]

where \( d_{gh} \) is the Poincar distance function defined in Eq. (1). The \( i \)-th label-aware document representation can be formed as the concatenation of all the similarities along the text sequence, i.e., \( s_i = [\text{score}(x_1, l_i); \ldots; \text{score}(x_T, l_i)] \). The set \( S = \{s_i\}_{i=1}^C \) acquired along the labels can be taken as the label-aware document representations under the hyperbolic word and label embeddings.

**Prediction** Given the document representations in \( S \), predictions can be made by a fully-connected layer and an output layer. The probability of each label for the document instance can be obtained by:

\[
p_i = \sigma(W^e\varphi(W^f s_i)), \forall s_i \in S, i = 1, \ldots, C,
\]

where \( \sigma \) is the sigmoid function, \( \varphi \) is a non-linearity. The weights \( W^e \in \mathbb{R}^{1 \times (T/2)} \) and \( W^f \in \mathbb{R}^{(T/2) \times C} \) are trainable parameters.

**Partial Interaction**

During the above interaction process, the amount of computation increases with the number of labels. When the output label space is large, it’s a burden to calculate the label-aware document representations. On account of the fact that only a few labels are assigned to one document instance, we propose to use a negative sampling method to improve the scalability during training. Let \( L^+ \) denote the set of true labels and \( L^- \) denote the set of randomly selected negative labels, the model is trained by minimizing the loss function which is derived from the binary cross-entropy loss as it is commonly used for MLC (Liu et al. 2017), i.e.,

\[
L^{b}_{\text{loss}} = - \left( \sum_{i \in L^+} \log(p_i) + \sum_{j \in L^-} \log(1 - p_j) \right).
\]

The hyperbolic parameters, i.e., \( \Theta^L \) and \( b \) in the hyperbolic word encoder, are updated via Riemannian adaptive optimization methods. The Euclidean parameters, i.e., \( W, U \) in the hyperbolic word encoder and \( W \) in the prediction layers, are updated via vanilla optimization methods. Partial interaction can significantly reduce the memory usage during training especially when the label set is large.
largest probability to be true, then the metrics are defined as

\[
    \text{Precision}@k = \frac{1}{k} \sum_{j=1}^{k} y|_{r[j]},
\]

\[
    \text{nDCG}@k = \frac{\sum_{i=1}^{k} y|_{r[i]}/\log(i+1)}{\sum_{i=1}^{\min(k,|y|)} 1/\log(i+1)},
\]

where \(|y|\) denotes the number of true labels, i.e., the number of 1 in \(y\). The final results are averaged over all the test document instances. Notice that nDCG@1 is omitted in the results since it gives the same value as P@1.

### Baselines

To demonstrate the effectiveness of HyperIM on the benchmark datasets, five comparative multi-label classification methods are chosen. EXAM (Du et al. 2019) is the state-of-the-art interaction model for text classification. EXAM uses pre-trained word embeddings in the Euclidean space, its label embeddings are randomly initialized. To calculate the similarity scores, EXAM uses the dot-product between word and label embeddings. SLEEC (Bhattachya et al. 2015) and DXML (Zhang et al. 2018) are two label-embedding methods. SLEEC projects labels into low-dimensional vectors which can capture label correlations by preserving the pairwise distance between them. SLEEC uses the k-nearest neighbors when predicting, and clustering is used to speed up its prediction. Ensemble method is also used to improve the performance of SLEEC. DXML uses DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) to embed the label co-occurrence graph into vectors, and uses neural networks to map the features into the embedding space. HR-DGCNN (Peng et al. 2018) and HMCN-F (Wehrmann, Cerri, and Barros 2018) are two neural network models specifically designed for hierarchical classification tasks.

### Experiments

#### Datasets

Experiments are carried out on three publicly available multi-label text classification datasets, including the small-scale RCV1 (Lewis et al. 2004), the middle-scale Zhihu, and the large-scale WikiLSHTC (Partalas et al. 2015). All the datasets are equipped with labels that explicitly exhibit a hierarchical structure. Their statistics can be found in Table 1.

| Dataset   | \(N_{\text{train}}\) | \(N_{\text{test}}\) | \(L\) | \(\bar{L}\) | \(W_{\text{train}}\) | \(W_{\text{test}}\) |
|-----------|-----------------|-----------------|-----|-----|----------------|----------------|
| RCV1      | 23,149          | 781,265         | 103 | 3.18| 729.67         | 259.47         |
| Zhihu     | 2,699,969       | 299,997         | 1,999| 2.32| 3513.17        | 38.14          |
| WikiLSHTC | 456,886         | 81,262          | 36,504| 1.86| 4.33           | 117.98         | 118.31

### Evaluation metrics

We use the rank-based evaluation metrics which have been widely adopted for multi-label classification tasks, i.e. Precision@k (P@k for short) and nDCG@k for \(k = 1, 3, 5\) (Bhattachya et al. 2015; Liu et al. 2017; Zhang et al. 2018). Let \(y \in \{0, 1\}^{|C|}\) be the ground truth label vector for a document instance and \(p \in [0, 1]^{|C|}\) be the predicted label probability vector. \(P@k\) records the fraction of correct predictions in the top \(k\) possible labels. Let the vector \(r \in \{1, \ldots, |C|\}^k\) denote the indices for \(k\) most possible labels in descending order, i.e. the \(r[1]\)-th label has the largest probability to be true, then the metrics are defined as

### Hyperparameters

To evaluate the baselines, hyperparameters recommended by their authors are used. EXAM uses the label embeddings dimension 1024 for RCV1 and Zhihu, on account of the scalability, it is set to 300 for WikiLSHTC. The word embedding dimension of EXAM is set to 300 for RCV1 and WikiLSHTC, 256 for Zhihu. SLEEC uses embedding dimension 100 for RCV1, 50 for Zhihu and WikiLSHTC. DXML uses embedding dimension 100 for RCV1, 300 for Zhihu and WikiLSHTC. The word embedding dimension of HR-DGCNN is 50 and its window size is set to be 5 to construct the graph of embeddings. Note that the original HMCN-F can’t take in the raw text data. To make HMCN-F more competitive, CNN-based architecture similar to XML-CNN (Liu et al. 2017) is adopted to extract the primary features. HMCN-F then fits its neural network layers to the label hierarchy, each layer focuses on predicting the labels in the corresponding hierarchical level.

### Experimental details

The hyperparameters used to pre-train the Poincar GloVe and the vanilla GloVe are the same values recommended by the authors. Partial routing is not applied when training HyperIM for RCV1, since the label set is not large. When evaluating HyperIM for WikiLSHTC, due to the scalability issue, the label set is divided into several groups. Models shared the same word embeddings, label embeddings and word encoder parameters predict different groups accordingly. This is feasible since the parameters in the prediction layers are the same for all the labels.

On account of the numeric error issue caused by the constraint \(|p| < 1\) for \(p \in \mathbb{S}^D\), when the embedding dimension \(k\) is large, the workaround is taken to address this issue, i.e. the embedding vector is a concatenation of vectors in the low-dimensional Poincar ball. Consequently, the embedding dimension for HyperIM is \(75 \times 2D\) as it generally outperforms the baselines.

---

https://biendata.com/competition/zhihu/
Numerical Errors  When the hyperbolic parameters go to the border of the Poincaré ball, gradients for the Mbius operations are not defined. Thus the hyperbolic parameters are always projected back to the ball with a radius \(1 - 10^{-5}\). Similarly when they get closer to \(0\), a small perturbation \((\epsilon = 10^{-15})\) is applied before they are used in the Mbius operations.

Optimization  The Euclidean parameters are updated via \(Adam\), and the hyperbolic parameters are updated via Riemannian adaptive \(Adam\) (Bécigneul and Ganea 2019). The learning rate is set to 0.001, other parameters take the default values. Early stopping is used on the validation set to avoid overfitting.

Results  As shown in Table 2, HyperIM consistently outperforms all the baselines. HyperIM effectively takes advantage of the label hierarchical structure comparing with \(EXAM\), \(SLEEC\) and \(DXML\). \(EXAM\) uses the interaction mechanism to learn word-label similarities, whereas clear connections between the words and the label hierarchy can’t be captured since its label embeddings are randomly initialized. The fact that HyperIM achieves better results than \(EXAM\) further confirms that HyperIM benefits from the retention of the hierarchical label relations. Meanwhile, the word embeddings learned by HyperIM have strong connections to the label structure, which is helpful to the measurement of word-label similarities and the acquirement of the label-aware document representations. \(SLEEC\) and \(DXML\) take the label correlations into account. However, the label correlations they use are captured from the label matrix, \(e.g.\) embedding the label co-occurrence graph, which may be influenced by tail labels. For HyperIM, the label relations are determined from the label hierarchy, so the embeddings of labels with parent-child relations are dependable to be correlated.

As expected, HyperIM is superior to the existing hierarchical classification methods HR-DGCNN and HMCN-F, even though they take advantage of the label hierarchy information. By investigating the properties of these three methods, we summarize the main reasons as follows. HR-DGCNN adds the regularization terms based on the assumption that labels with parent-child relations should have similar weights in the fully-connected layer, which may not always be true in real applications. HMCN-F highly depends on the label hierarchy, it assumes that different paths pass through the same number of hierarchical levels. Unfortunately, in the real data, different paths may have totally different lengths. HyperIM models the label relations by embedding the label hierarchy in the hyperbolic space. Any hierarchical structure can be suitable and labels aren’t required to sit on a specific hierarchical level, which makes HyperIM less reliant on the label hierarchy. Furthermore, HyperIM can learn the word-label similarities and preserve the label relations simultaneously to acquire label-aware document representations, whereas HR-DGCNN and HMCN-F treat document words and labels separately.

Ablation Test  In order to show the characteristics of HyperIM and justify the superiority of the hyperbolic space for hierarchical multi-label text classification, we are interested in comparing it with an analogous model in the Euclidean space.

Euclidean Interaction Model  The analogous model in the Euclidean space (EuclideanIM) has a similar architecture as HyperIM. EuclideanIM takes the vanilla pre-trained GloVe word embeddings (Pennington, Socher, and Manning 2014) and uses the vanilla \(GRU\) (Chung et al. 2014) as the word encoder. The label embeddings are randomly initialized for \(E-rand\), while \(E-hier\) takes the same label embeddings initialized by the hierarchical label relations as \(H-hier\). The word-label similarities are computed as the negative of the Euclidean distance between their embeddings, \(i.e.\) \(score(x_i, l) = -\|\theta^w_i - \theta^l_i\|\) for \(\theta^w_i, \theta^l_i \in \mathbb{R}^k\). The same architecture of the prediction layers is adopted.

Results  Figure 3 shows the \(nDCG@k\) results for different embedding dimensions on the RCV1 dataset. The fact that \(E-hier\) slightly outperforms \(E-rand\) indicates that the label correlations provide useful information for classification. However, \(H-rand\) still achieves better results than the

### Table 2: Results in \(P@k\) and \(nDCG@k\), bold face indicates the best in each line.

| Dataset | Metrics | EXAM | SLEEC | DXML | HR-DGCNN | HMCN-F | HyperIM |
|---------|---------|------|-------|------|----------|--------|---------|
| RCV1    | \(P@1\) | 95.98| 94.45 | 95.27| 95.17    | 95.35  | 96.78   |
|         | \(P@3\) | 80.83| 78.60 | 77.86| 80.32    | 78.95  | 81.46   |
|         | \(P@5\) | 55.80| 54.24 | 53.44| 55.38    | 55.90  | 56.79   |
|         | \(nDCG@3\) | 90.74| 90.05 | 89.69| 90.02    | 90.14  | 91.52   |
|         | \(nDCG@5\) | 91.26| 90.32 | 90.24| 90.28    | 90.82  | 91.89   |
| Zhihu   | \(P@1\) | 51.41| 51.34 | 50.34| 50.97    | 50.24  | 52.14   |
|         | \(P@3\) | 32.81| 32.56 | 31.21| 32.41    | 32.18  | 33.66   |
|         | \(P@5\) | 24.29| 24.23 | 23.36| 23.87    | 24.09  | 24.99   |
|         | \(nDCG@3\) | 49.32| 49.27 | 47.92| 49.02    | 48.36  | 50.13   |
|         | \(nDCG@5\) | 50.74| 49.71 | 48.65| 49.91    | 49.21  | 51.05   |
| WikiLSHTC | \(P@1\) | 54.90| 53.57 | 52.02| 52.67    | 53.23  | 55.06   |
|         | \(P@3\) | 30.50| 31.25 | 30.57| 30.13    | 29.32  | 31.73   |
|         | \(P@5\) | 22.02| 22.46 | 21.66| 22.85    | 22.19  | 23.08   |
|         | \(nDCG@3\) | 49.50| 49.27 | 47.92| 49.02    | 48.36  | 50.13   |
|         | \(nDCG@5\) | 50.46| 47.52 | 48.14| 50.42    | 49.87  | 51.36   |
Euclidean models even without the hierarchical label structure information, which confirms that the hyperbolic space is more suitable for HMLC. For E-hier, the hierarchical label structure is not appropriate to be embedded in the Euclidean space, thus it can’t fully take advantage of such information. HyperIM generally outperforms EuclideanIM and achieves significant improvement especially in low-dimensional latent space. H-hier takes in the label correlations and outperforms H-rand as expected.

Interaction Visualization  The 2-dimensional hyperbolic label embeddings and the encoded word embeddings (not the pre-trained word embeddings) can be visualize jointly in the Poincar disk as shown in Figure 4. The hierarchical label structure which can represent the parent-child relations between labels is well preserved by HyperIM. Note that the embedded label hierarchy resembles the embedded tree in Figure 1b. The top-level nodes (e.g. the label node A) are embedded near the origin of the Poincar disk, while the leaf nodes are close to the boundary. The hierarchical label relations are well modeled by such tree-like structure. Moreover, in the dataset, the top-level labels are not connected to an abstract ”root”. The structure of the embedded label hierarchy still suggests that there should be a ”root” that connects all the top-level labels to put at the very origin of the Poincar disk, which indicates that HyperIM can really make use of the hierarchical label relations.

The explicit label correlations can further help HyperIM to learn to encode the word embeddings via interaction. The encoded text of a document instance are generally embedded close to the assigned labels. This clear pattern between the encoded word embeddings and the label hierarchy indicates that HyperIM learns the word-label similarities with the label correlations taken into consideration. This is the main reason that HyperIM outperforms EuclideanIM significantly in low dimensions. Some of the words such as ”the”, ”is” and ”to” don’t provide much information for classification, putting these words near the origin can make them equally similar to labels in the same hierarchical level. A nice by-product is that the predicted probabilities for labels in the same hierarchical level wont be influenced by these words. Moreover, the variance of word-label distance for labels in different hierarchical levels make parent labels distinguishable from child labels, e.g. top-level labels can be made different from the leaf labels since they are generally closer to the word embeddings. Such difference suggests that HyperIM treats the document instances differently along the labels in different hierarchical levels.

Embedding Visualization  The word/label embedding in the 2-dimensional latent space obtained by H-hier and E-hier are demonstrated respectively in Figure 5. As expected, the hierarchical label structure which can represent the parent-child relations between labels is well preserved by HyperIM (as shown in Figure 5a), while the label embeddings in the Euclidean space are less interpretable as the paths among labels intersect each other. Consequently, it is hard to find the hierarchical label relations in the Euclidean space. Moreover, the word embeddings are generally near the origin, which gives the word encoder the opportunities to make a piece of text of a document instance tend to different directions where the true labels locate. Whereas it’s more difficult to adjust the word embeddings in the Euclidean space since different label hierarchies are not easy to distinguish.
Related Work

Hierarchical Multi-Label Classification

The existing methods dedicating to hierarchical classification usually focus on the design of loss functions or neural network architectures (Cerri, Barros, and de Carvalho 2015). Traditional hierarchical classification methods optimize a loss function locally or globally (Silla and Freitas 2011). Local methods are better at capturing label correlations, whereas global methods are less computationally expensive. Researchers recently try to use a hybrid loss function associated with specifically designed neural networks, e.g. HR-DGCNN (Peng et al. 2018). The archetype of HMCN-F (Wehrmann, Cerri, and Barros 2018) employs a cascade of neural networks, where each neural networks layer corresponds to one level of the label hierarchy. Such neutral network architectures generally require all the paths in the label hierarchy to have the same length, which limits their application. Moreover, on account of the fact that labels in high hierarchical levels usually contain much more instances than labels in low levels, whereas neural network layers for low levels need to classify more labels than layers for high levels, such architectures also lead to imbalance classification.

Hyperbolic Deep Learning

Research on representation learning (Nickel and Kiela 2017) indicates that the hyperbolic space is more suitable for embedding symbolic data with hierarchical structures than the Euclidean space, since the tree-likeness properties (Hamann 2018) of the hyperbolic space make it efficient to learn hierarchical representations with low distortion (Sarkar 2011). Since linguistic ontologies are innately hierarchies, hierarchies are ubiquitous in natural language, (e.g. WordNet (Miller 1995)). Some works lately demonstrate the superiority of the hyperbolic space for natural language processing tasks such as textual entailment (Ganea, Béginneul, and Hofmann 2018), machine translation (Gulcehre et al. 2019) and word embedding (Tifrea, Béginneul, and Ganea 2019).

Riemannian optimization

In the same way that gradient-based optimization methods are used for trainable parameters in the Euclidean space, the hyperbolic parameters can be updated via Riemannian adaptive optimization methods (Béginneul and Ganea 2019). For instance, Riemannian adaptive SGD updates the parameters $\theta \in B^k$ by $\theta_{t+1} = \exp_{\theta_t}(-\eta \nabla_R \mathcal{L}(\theta_t))$, where $\eta$ is the learning rate, and the Riemannian gradient $\nabla_R \mathcal{L}(\theta_t) \in T_{\theta}B^k$ is the rescaled Euclidean gradient, i.e. $\nabla_R \mathcal{L}(\theta) = \frac{1}{\lambda^2} \nabla_E \mathcal{L}(\theta)$ (Wilson and Leimeister 2018).

Conclusion

The hierarchical parent-child relations between labels can be well modeled in the hyperbolic space. The proposed HyperIM is able to explicitly learn the word-label similarities by embedding the words and labels jointly and preserving the label hierarchy simultaneously. HyperIM acquires label-aware document representations to extract the fine-grained text content along each label, which significantly improves the hierarchical multi-label text classification performance. Indeed, HyperIM makes use of the label hierarchy, whereas there is usually no such hierarchically organized labels in practice, especially for extreme multi-label classification (XMLC). Nevertheless, the labels in XMLC usually follow a power-law distribution due to the amount of tail labels (Babbar and Schölkopf 2018), which can be traced back to hierarchical structures (Ravasz and Barabási 2003). Thus, it will be interesting to extend HyperIM for XMLC in the future. Our code is publicly available to facilitate other research.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61822601, 61773050, and 61632004; the Beijing Natural Science Foundation under Grant Z180006; the Beijing Municipal Science & Technology Commission under Grant Z181100008918012; National Key Research and Development Program (2017YFC1703506); the Fundamental Research Funds for the Central Universities (2019JBZ110).
