Hyperbolic Deep Keyphrase Generation

Yuxiang Zhang[1](✉), Tianyu Yang[1], Tao Jiang[1], Xiaoli Li[2], and Suge Wang[3]

[1] Civil Aviation University of China, Tianjin, China
{yxzhang,2019051011,2020052049}@cauc.edu.cn
[2] Institute for Infocomm Research/Centre for Frontier AI Research, Singapore
xlli@i2r.a-star.edu.sg
[3] Shanxi University, Taiyuan, China
wsg@sxu.edu.cn

Abstract. Keyphrases can concisely describe the *high-level topics* discussed in a document, and thus keyphrase prediction compresses document's hierarchical semantic information into a few important representative phrases. Numerous methods have been proposed to use the encoder-decoder framework in Euclidean space to generate keyphrases. However, their ability to capture the hierarchical structures is limited by the nature of Euclidean space. To this end, we propose a new research direction that aims to encode the hierarchical semantic information of a document into the low-dimensional representation and then decompress it to generate keyphrases in a *hyperbolic space*, which can effectively capture the underlying semantic hierarchical structures. In addition, we propose a novel *hyperbolic attention mechanism* to selectively focus on the high-level phrases in hierarchical semantics. To the best of our knowledge, this is the first study to explore a hyperbolic network for keyphrase generation. The experimental results illustrate that our method outperforms fifteen state-of-the-art methods across five datasets.

Keywords: Keyphrase generation · Hyperbolic neural network · Hyperbolic attention mechanism.

1 Introduction

Keyphrase prediction is to automatically produce a set of representative phrases that are related to the main topics discussed in a given document. Since keyphrases (also referred to as keywords) can provide a *high-level topic description* of a document, they are beneficial for a wide range of natural language processing (NLP) tasks, such as information extraction [32], text summarization [33] and question generation [30]. However, the performance of existing approaches is still far from being satisfactory [21,16]. The main reason is that it is very challenging to determine if a phrase or a set of phrases accurately capture the high-level topics that are presented in a document.

Automatic keyphrase prediction models can be broadly divided into *extraction* and *generation* methods. In particular, traditional *extraction methods* can

only extract *present keyphrases* that appear in a given document, while *generation methods* can generate both present keyphrases as well as *absent keyphrases* that do not appear in a document.

Recently, the sequence-to-sequence (seq2seq) framework has been widely applied in the natural language generation tasks. CopyRNN [23] is the first to employ the attentional seq2seq framework [29] with the copying mechanism [14] to generate both present and absent keyphrases for a document. Following CopyRNN, several seq2seq-based keyphrase generation methods have been proposed to improve the generation performance [6,36,41,34,8,38,1,37]. However, all these existing keyphrase generation methods have been proposed to compress the semantic information in a given document into a dense vector in Euclidean space, assuming a *flat* geometry. Although these Euclidean representation models have proved successful for the keyphrase generation task, they still suffer from an inherent limitation: their ability to capture *hierarchical structures* is bounded by the nature of flat geometry of Euclidean space, as mentioned in recent work [27].

As a given document covers different topics and consists of many phrases which could be keyphrases, it is critical to represent it into a *hierarchical semantic representation*, facilitating the selection of the most representative keyphrases related to the main topics at the highest level. Fig.1 shows the hierarchical relations among different semantic levels of candidate keyphrases, which can be regarded as the ideal keyphrase generation if viewing it from low-level (boundary) to high-level (center) candidates. In Fig.1, the set of ideal keyphrases should be KP = $\{\text{cp}_1, \text{cp}_2, \text{cp}_3\}$ at the highest level, covering three topics comprehensively. If the set of predicted keyphrases is KP′=$\{\text{cp}_{21}, \text{cp}_{221}\}$, it just provides a *local and low-level* topic description of the second topic Topic_2 only, ignoring the other two topics and a part of the second topic. This example illustrates that without an effective hierarchical semantic representation, the predicted kephrases will not cover major topics and provide the high-level topic description. As mentioned in several existing studies [21,16,6,41,38], predicted keyphrases may fall into a single topic and fail to cover all the main topics discussed in a document. In summary, semantic hierarchical relations widely exist among keyphrases, but existing keyphrase generation methods available in *Euclidean space* can not effectively capture *semantic hierarchical relations* to improve the topic coverage of predicted keyphrases.

Recently, hyperbolic representation methods [27,26] have been developed to model the latent hierarchical nature of data and demonstrated encouraging results. To efficiently utilize hyperbolic embeddings in downstream tasks, researchers have proposed some advanced hyperbolic deep networks, such as hyperbolic neural networks [12] and hyperbolic attention network [15].

Motivated by the above observations, we propose a *hyperbolic seq2seq network* for keyphrase generation, which is a novel keyphrase generation framework for modeling hierarchical relations. Specifically, we design a *hyperbolic encoder* to compress the hierarchical semantic information discussed in a target document into a hyperbolic embedding, and devise a *hyperbolic decoder* to generate corresponding keyphrases. In the hyperbolic network, we propose an innova-

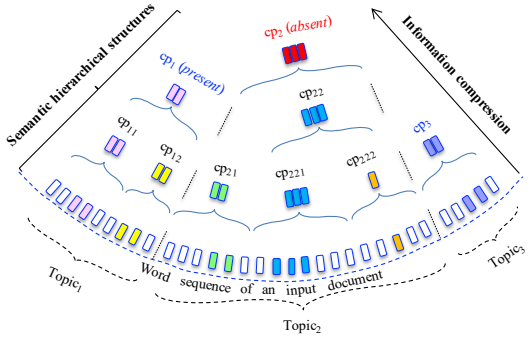

Fig. 1. Ideal semantic hierarchical relations among candidate keyphrases (cp) within a document, in which the dotted line semantically represents a topic segmentation and facilitates understanding of hierarchical structures of topics.

tive *hyperbolic hierarchy-aware attention mechanism* to enhance the ability to learn semantic hierarchical relations, which can selectively focus on the words with high-level semantics. Different from Euclidean deep generation methods, our proposed hyperbolic hierarchy-aware attention mechanism make our model more effective to capture the semantic hierarchical relations within a target document and thus generate keyphrases based on its semantic understanding with good topic coverage and accuracy. In addition, we propose a new *metric* to measure the degree to which the predicted keyphrases cover the main topics of a target document. To the best of our knowledge, this is the first work to design a new *hyperbolic network* for keyphrase generation.

2 Related Work

2.1 Keyphrase Generation

Following CopyRNN [23], several extensions have been proposed to boost its generation ability. For instance, Ye et al. [36] propose a semi-supervised keyphrase generation model that utilizes both abundant unlabeled data and limited labeled data. Chen et al. [9] propose a title-guided network to sufficiently utilize the already summarized information in given title. In addition, some researches attempted to leverage external knowledge to help reducing duplication and improving coverage, such as syntactic constraints [41] and latent topics [34].

The above-mentioned methods, which utilize the standard seq2seq network, can not generate multiple keyphrases and determine the appropriate number of keyphrases at a time for a target document. To overcome this shortcoming, Yuan et al. [38] introduce the new training and inference setup in the seq2seq network to generate multiple keyphrases and decide on the suitable number of keyphrases for a given document. Ye et al., [37] propose a One2Set paradigm to predict the keyphrases as a set, which eliminates the bias caused by the predefined order

in One2Seq paradigm [38]. In addition, some recent works focus on improving the decoding process of seq2seq networks. For example, Chen et al., [8] propose an exclusive hierarchical decoding framework and use either a soft or a hard exclusion mechanism to reduce duplicated keyphrases. More recently, Ahmad et al. [1] design an extractor-generator to jointly extract and generate keyphrases from a document. We observe that almost all existing keyphrase generation methods used the Euclidean seq2seq framework, which cannot provide the most powerful representations for hierarchical structures on keyphrase generation task.

2.2 Hyperbolic Representation

An increasing number of research has shown that many types of complex data exhibit non-Euclidean structures [3]. Recently, hyperbolic embedding methods have been proposed to learn the latent representation of hierarchical data and demonstrated encouraging results. In the field of NLP, hyperbolic representation learning has been successfully applied to generating word embeddings [31] and sentence representations [10], and inferring concept hierarchies from large text corpora [20]. In addition, hyperbolic geometry has been integrated into recent advanced hyperbolic deep learning frameworks, such as hyperbolic neural networks [12], and hyperbolic attention network [15].

3 Preliminaries

Hyperbolic space Hyperbolic space, specifically referring to a simply connected manifolds with constant negative curvature [2], can be thought of as a continuous analogue of tree and is more suitable for learning data with hierarchical structures. The hyperbolic space can be constructed using various isomorphic models (*i.e.*, these models can be converted into each other). In this paper, we follow the majority of NLP works and employ the Poincaré ball model with the curvature set as -1, whose distance function is differentiable.

Poincaré ball model The n-dimensional *Poincaré ball model* $\mathcal{P}^n = (\mathcal{B}^n, g^{\mathcal{P}})$ is defined by a Riemannian manifold $\mathcal{B}^n = \{\mathbf{x} \in R^n \mid \|\mathbf{x}\| < 1\}$ with the metric tensor $g^{\mathcal{P}}(\mathbf{x}) = (\frac{2}{1-\|\mathbf{x}\|^2})^2 g^{\mathcal{E}}$, where $\|\cdot\|$ denotes the Euclidean norm, and $g^{\mathcal{E}} = \mathbf{I}_n$ is the Euclidean metric tensor. The *induced distance* between two points $\mathbf{x}, \mathbf{y} \in \mathcal{P}^n$ is defined as

$$d_{\mathcal{P}}(\mathbf{x}, \mathbf{y}) = \cosh^{-1}\left(1 + \frac{2\|\mathbf{x}-\mathbf{y}\|^2}{(1-\|\mathbf{x}\|^2)(1-\|\mathbf{y}\|^2)}\right), \tag{1}$$

where $\cosh^{-1}(x) = \ln(\mathrm{x} + \sqrt{\mathrm{x}^2 - 1})$ is an inverse hyperbolic cosine function.

The induced distance can place root node near the center of the ball and leaf nodes near the boundary of the ball to ensure that the distance from the root node to each of leaf nodes is relatively small while the distance between leaf nodes is relatively large. This explains why hyperbolic space can be seen as a tree-like hierarchical structure.

Klein model To define the *hyperbolic average*, we employ the Klein model of hyperbolic space. The n-dimensional *Klein model* $\mathcal{K}^n = (\mathcal{B}^n, g^{\mathcal{K}})$ is also defined in a manifold $\mathcal{B}^n$ with the different metric tensor $g^{\mathcal{K}}$. The Poincaré model and Klein model describe the same hyperbolic space using different coordinates. Thus, these two models can be converted into each other. Given a point $\mathbf{x}^{\mathcal{P}} \in \mathcal{P}^n$ in the Poincaré ball model, we convert it to the Klein model by $\mathbf{x}^{\mathcal{K}} = \frac{2\mathbf{x}^{\mathcal{P}}}{1+\|\mathbf{x}^{\mathcal{P}}\|^2}$. Similarly, a point $\mathbf{x}^{\mathcal{K}} \in \mathcal{K}^n$ in Klein model can be converted into Poincaré ball model as $\mathbf{x}^{\mathcal{P}} = \frac{\mathbf{x}^{\mathcal{K}}}{1+\sqrt{1-\|\mathbf{x}^{\mathcal{K}}\|^2}}$.

Hyperbolic operations To make neural networks work in hyperbolic space, Möbius operations including *Möbius addition* and *Möbius matrix-vector multiplication* in the Poincaré ball are used. In addition, the *exponential map* (which maps a Euclidean vector to the hyperbolic space) and the inverse *logarithm map* are also used. The details of these operations can be seen in the work [12].

4 Methodology

4.1 Problem Definition

Let $x = (x_1, \ldots, x_T)$ be a document that is treated as a sequence of words, where T is the length of x. The goal of a keyphrase generation method is to find a model to generate a set of keyphrases $K = \{p_k\}_{k=1}^{|K|}$ for document x, where each keyphrase $p_k = (y_1, ..., y_{|p_k|})$ is also a sequence of words.

To generate multiple keyphrases for an input document, existing approaches provide two different data formats as the predicted keyphrase output (*i.e.*, two training paradigms): *One2One* [23] and *One2Seq* [38]. One2One only predicts a fixed number of keyphrases for all documents, where each training data sample is a pair of source text and one of its keyphrases (x, p). To overcome this drawback, One2Seq can generate a single sequence, which consists of multiple predicted keyphrases and separators, as represented by $K' = p_1$ <sep> p_2... <sep> $p_{|K|}$. Each training data sample is a pair of source text and concatenated sequence of its keyphrases and separators (x, K').

4.2 Hyperbolic Encoder-Decoder Model

The basic idea of our keyphrase generation model is to leverage a *hyperbolic deep network* to compress the semantic information of the input document into the low-dimensional representations using the hyperbolic encoder and to generate corresponding keyphrases using the hyperbolic decoder, based on the representations. In this hyperbolic network, we propose a new *hyperbolic attention* mechanism to capture the semantic subordination and select the words with high-level semantics. In addition, a hyperbolic pointer mechanism is used to copy certain out-of-vocabulary words from the input document and paste them into the generated keyphrases. An overview of this method is shown in Fig.2.

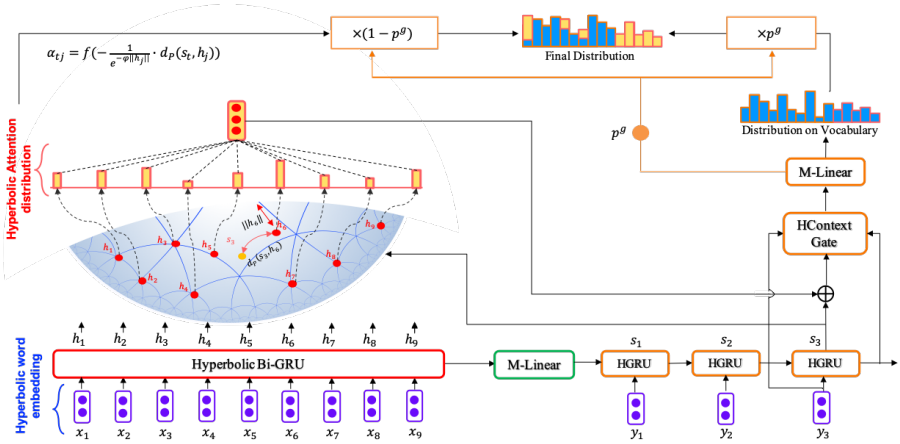

Fig. 2. The overview of the proposed hyperbolic deep model for keyphrase generation.

The encoder and decoder are implemented with a hyperbolic gated recurrent unit (HGRU) [12]. Let $x = (x_1, ..., x_T)$ be a sequence of words within an input document, and $\mathbf{x} = [\mathbf{x}_1, ..., \mathbf{x}_T]$ be its corresponding sequence of hyperbolic word embeddings. The encoder maps the input word sequence x into a set of contextualized hidden representations $\mathbf{h} = [\mathbf{h}_1, ..., \mathbf{h}_T]$, using a bidirectional HGRU $\mathbf{h}_t = [\mathrm{HGRU_f}(\mathbf{x}_t), \mathrm{HGRU_b}(\mathbf{x}_t)]$ where $\mathrm{HGRU_f}(\cdot)$ and $\mathrm{HGRU_b}(\cdot)$ are used to learn the forward and backward hidden states around the input text, respectively. HGRU based on Möbius operations in Poincaré model [12] is defined as

$$\begin{aligned}
\mathbf{r}_t &= \sigma(\log_0(\mathbf{W}^r \otimes \mathbf{h}_{t-1} \oplus \mathbf{U}^r \otimes \mathbf{x}_t \oplus \mathbf{b}^r)) \\
\mathbf{z}_t &= \sigma(\log_0(\mathbf{W}^z \otimes \mathbf{h}_{t-1} \oplus \mathbf{U}^z \otimes \mathbf{x}_t \oplus \mathbf{b}^z)) \\
\tilde{\mathbf{h}}_t &= \varphi((\mathbf{W}^h \mathrm{diag}(\mathbf{r}_t)) \otimes \mathbf{h}_{t-1} \oplus \mathbf{U}^h \otimes \mathbf{x}_t \oplus \mathbf{b}^h) \\
\mathbf{h}_t &= \mathbf{h}_{t-1} \oplus \mathrm{diag}(\mathbf{z}_t) \otimes (-\mathbf{h}_{t-1} \oplus \tilde{\mathbf{h}}_t)
\end{aligned} \tag{2}$$

where $\mathbf{r}_t$ is a reset gate, and $\mathbf{z}_t$ is a update gate. $\log_0(\cdot)$, $\otimes$ and $\oplus$ are defined in subsection 3. $\sigma(\cdot)$ is a sigmoid function, and $\varphi(\cdot)$ is a pointwise non-linearity. Since the hyperbolic space naturally has non-linearity, $\varphi(\cdot)$ is identity. $\mathrm{diag}(\cdot)$ is a square diagonal matrix. The six weights $\mathbf{W} \in R^{n \times n}$, $\mathbf{U} \in R^{n \times m}$ are trainable parameters in Euclidean space and three biases $\mathbf{b} \in \mathcal{B}^n$ are trainable parameters in hyperbolic space.

The decoder is another forward HGRU which is used to generate the sequence of keyphrases by predicting the next word y_t based on the hidden state $\mathbf{s}_t$. Both y_t and $\mathbf{s}_t$ are conditioned on $\mathbf{y}_{t-1}$ and $\mathbf{c}_t$ of the input sequence. Formally, the hidden state $\mathbf{s}_t$ and decoding function can be written as

$$\mathbf{s}_t = \mathrm{HGRU_f}(\mathbf{y}_{t-1}, \mathbf{s}_{t-1}, \mathbf{c}_t), \tag{3}$$

and

$$p(y_t \mid y_1, y_2, ..., y_{t-1}, \mathbf{c}) = g(\mathbf{y}_{t-1}, \mathbf{s}_t, \mathbf{c}_t), \tag{4}$$

where $g(\cdot)$ is a nonlinear multi-layered function that outputs the probability of y_t. The more details of the decoder are given in the next subsections.

4.3 Hyperbolic Attention Mechanism

The attention mechanism is used to make the network model dynamically focus on the important parts in input data, and consists of two core parts: matching and aggregation. Particularly, the matching part computes attention weight $\alpha_{tj} = \alpha(\mathbf{s}_t, \mathbf{h}_j)$, which reflects the relevance of the hidden states $\mathbf{h}_j$ of input sequence in the presence of the current hidden state $\mathbf{s}_t$ for deciding the next word y_t. The aggregation part, on the other hand, takes a weighted sum of hidden states using these weights, also known as context vector $\mathbf{c}_t$.

A general hyperbolic attention mechanism was first introduced by Gulcehre et al. [15] to build an attentive read operation in the Hyperboloid model. Inspired by this work, we propose a new hyperbolic attention mechanism in the Poincaré ball model *specifically* for the keyphrase generation task. In particular, in the matching part, the most natural way to compute attention weight is to use the hyperbolic distance between points of matching pairs, given as $\alpha_{tj} = \frac{\exp(a(\mathbf{s}_t, \mathbf{h}_j))}{\sum_{k=1}^{T} \exp(a(\mathbf{s}_t, \mathbf{h}_k))}$, where $a(\mathbf{s}_t, \mathbf{h}_j)$ is a soft alignment function that is used to score how well the inputs around position j and the output at position t match (*i.e.*, to measure the relevance between $\mathbf{s}_t$ and $\mathbf{h}_j$), computed as

$$a(\mathbf{s}_t, \mathbf{h}_j) = -\beta d_{\mathcal{P}}(\mathbf{s}_t, \mathbf{h}_j) - d_b, \tag{5}$$

where $d_{\mathcal{P}}(\cdot, \cdot)$ is the distance function in hyperbolic space, and d_b is a parameter learned along with the rest of the network. Note that in the work [15], β is also a learnable coefficient. This causes the attention mechanism to only utilize the hyperbolic distance to measure the relevance between $\mathbf{s}_t$ and $\mathbf{h}_j$, and *ignore* the distance between the center of the Poincaré ball to $\mathbf{h}_j$ (*i.e.*, the norm of $\mathbf{h}_j$), which can reflect the semantic level of an input word at position j (*i.e.*, hidden state $\mathbf{h}_j$) in a tree-like hierarchical structure internalized by the hyperbolic space.

To overcome this drawback and further enhance the ability to capture the high-level semantics, we redefine β as

$$\beta = \frac{1}{\exp(-\varphi \|\mathbf{h}_j\|)}, \tag{6}$$

where φ is a hyper parameter. Thus, the new hyperbolic attention mechanism takes into account not only the semantic relevance between two words but also the *semantic hierarchy* of each word in the semantic tree (as Fig.1 shows). We name it as the hierarchy-aware attention mechanism.

In the aggregation part, the weighted sum of hidden states is computed by the Einstein midpoint that is defined in Klein model as $\mathbf{c}_t = \sum_{j=1}^{T} \left[\frac{\alpha_{tj}\gamma(\mathbf{h}_j)}{\sum_{l=1}^{T} \alpha_{tl}\gamma(\mathbf{h}_l)} \right] \mathbf{h}_j$, where $\gamma(\mathbf{h}_j) = \frac{1}{\sqrt{1-\|\mathbf{h}_j\|^2}}$ is a Lorentz factor.

Note that before aggregation process, we first transform the hidden states from Poincaré Ball to Klein model, and transform it back to Poincaré ball model after aggregation. The used formulas are given in subsection Klein model.

4.4 Hyperbolic Pointing Mechanism

To recall some keyphrases which contain out-of-vocabulary words, CopyRNN utilized the copying mechanism [14] to generate out-of-vocabulary words. Here, we use the pointing mechanism (that is a modified copy mechanism) into the Poincaré ball model for the same purpose.

Let $\mathcal{V}$ be a global vocabulary, $\mathcal{V}_s$ be a vocabulary of the source sentences, and *unk* be any out-of-vocabulary word. It builds an extended vocabulary $\mathcal{V}_e = \mathcal{V} \cup \mathcal{V}_s \cup \{unk\}$. The distribution over $\mathcal{V}_e$ at current time step t is

$$p(y_t) = p_t^g \cdot p_g(y_t) + (1 - p_t^g) \cdot p_c(y_t), \tag{7}$$

where p_t^g is the probability of choosing generate-mode, calculated by

$$p_t^g = \sigma(\log_0(\mathbf{W}^{cg} \otimes \mathbf{d}_t \oplus \mathbf{b}^{cg})). \tag{8}$$

The probability of generate-mode $p_g(\cdot)$ and copy-mode $p_c(\cdot)$ are given by

$$p_g(y_t) = \begin{cases} \mathbf{v}_i^\top \log_0(\mathbf{W}^g \otimes \mathbf{d}_t \oplus \mathbf{b}^g), & y_t \in \mathcal{V} \cup \{unk\}, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

$$p_c(y_t) = \begin{cases} \sum_{j:x_j=y_t} \alpha_{tj}, & y_t \in \mathcal{V}_s, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

where $\mathbf{v}_i$ is a one-hot indicator vector, $\mathbf{W}$ and $\mathbf{b}$ in Eq. (8) and Eq. (9) are trainable parameters. Finally, we adopt the widely used cross entropy loss function to train the models, both in One2One and One2Seq paradigms.

5 Experiments

5.1 Dataset

We employ KP20k dataset [23], where each example contains a title and an abstract of a scientific paper as source text, and author-assigned keywords as target keyphrases. Following previous works, we use the training dataset of KP20k to train all the models, and use the validation dataset to validate the choice of hyper parameters. In order to evaluate the proposed model comprehensively, we also test on other four widely used public datasets from the scientific domain, namely, Inspec [17], Krapivin [19], SemEval-2010 [18] and NUS [25]. The detailed statistic information of these five datasets are summarized in the work [40].

5.2 Baselines

For the *present* keyphrase prediction, we compare our models with two types of methods, including eight extraction and seven deep generation methods.

Representative *extraction* methods consist of three different types: 1) *statistic-based* unsupervised methods, including (1) **TF-IDF** and (2) **YAKE!** [4], 2) *graph-based* unsupervised methods, including (3) **TextRank** [24], (4) **SingleRank** [32], (5) **PositionRank** [11], (6) **KPRank** [28], and 3) traditional supervised methods, including (7) **KEA** [35] and (8) **Maui** [22]. Due to the limited space, we select the *best-performing method* (BL*) from *each of the three types of baselines* with the best-performing metrics to compare with our method.

The supervised *generation* baselines can be classified into *One2One* and *One2Seq* according to the training paradigm. The One2One baselines include: (1) **CopyRNN** [23], which is the first to use seq2seq network to generate keyphrases, (2) **CorrRNN** [6], which is an extension of CopyRNN integrating the sequential decoding with coverage and review mechanisms, and (3) **KG-KE-KR-M** (abbreviated as **KG-KE**) [7], which is a multi-task learning using extraction and generation models to generate keyphrases.

The One2Seq baselines include: (1) **CatSeq** [38], which has the same framework as CopyRNN, with the key difference between them on the training paradigm, (2) **CatSeqTG-2RFl** (abbreviated as **Cat-2RFl**) [5], which is an extension of CatSeq using reinforcement learning to generate both sufficient and accurate keyphrases, (3) **ExHiRD-h** [8], which uses an exclusive hierarchical decoder to avoid generating duplicated keyphrases, and (4) **SEG-Net** [1], which jointly extracts and generates keyphrases.

The proposed **Hy**perbolic **A**ttentional **N**etwork (HyAN[1]) and its extensions are: (1) HyAN, which is a basic hyperbolic attentional model trained by One2One paradigm, corresponding to CopyRNN, (2) HyAN$_h$, which is an extension of HyAN, in which only the semantic hierarchy is integrated into the hyperbolic attention mechanism, (3) HyANS, which is also an extension of HyAN trained by One2Seq paradigm, corresponding to CatSeq, and (4) HyANS$_h$, which is a composite of HyANS and HyAN$_h$, trained by One2Seq paradigm and incorporated with the semantic hierarchy.

5.3 Evaluation Metrics

We adopt top-N macro-averaged *F-measure* (F_1) and $R@k$ as the evaluation metrics, in which F_1 includes $F_1@k$, $F_1@\mathcal{O}$ and $F_1@\mathcal{M}$. $F_1@k$ is used in almost all existing works, while $F_1@\mathcal{O}$ and $F_1@\mathcal{M}$ proposed in [38] are designed specifically for the One2Seq generation, where $\mathcal{O}$ is the number of author-provided keyphrases and $\mathcal{M}$ is the number of all predicted keyphrases. They are capable of reflecting the nature of variable number of keyphrases for each document. The recall of the top 50 predictions ($R@50$) evaluates prediction of absent keyphrases.

5.4 Implementation Details

We follow the previous works [23,38] to pre-process the experimental data. The top 50,000 most frequently-occurring words in the training data are used as the vocabulary shared in the hyperbolic encoder and decoder.

[1] The code of our model is available at `https://github.com/SkyFishMoon/HyAN`.

Table 1. Results of predicting *present* keyphrases of different methods on five datasets. Best/second-best performing score in each column is highlighted with bold/underline in each of two trained paradigms, and best performing score in both trained paradigms is highlighted with bold and asterisk. CopyRNN$^+$ is re-implemented CopyRNN with best results [38]. Sta-, Gra- and Tra- represent statistic-unsupervised, graph-unsupervised and traditional supervised, respectively.

	Method	Inspec		Krapivin		NUS		SemEval		KP20k	
		F_1@5	F_1@10	F_1@5	F_1@10	F_1@5	F_1@10	F_1@5	F_1@10	F_1@5	F_1@10
Ext.	Sta-BL*	20.4	24.4	21.5	19.6	15.9	19.6	15.1	21.2	14.1	14.6
	Gra-BL*	27.7	32.3	17.7	18.5	21.0	22.3	22.5	25.7	18.1	15.0
	Tra-BL*	10.9	12.9	24.3	20.8	24.9	26.1	4.50	3.90	26.5	22.7
One2One	CopyRNN$^+$	24.4	28.9	30.5	26.6	37.6	35.2	31.8	31.8	31.7	27.3
	CorrRNN	-	-	31.8	27.8	35.8	33.0	32.0	32.0	-	-
	KG-KE	25.7	28.4	27.2	25.0	28.9	28.6	20.2	22.3	31.7	28.2
	HyAN	27.9	29.8	32.2	27.9	38.1	34.7	32.8	32.3	32.9	28.5
	HyAN$_h$	**28.8**	**30.2**	**33.0**	**28.9**	**38.8**	**36.2**	**33.3***	**32.5**	**34.0**	**29.3**
		F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$
One2Seq	CatSeq	22.5	26.2	26.2	35.4	32.3	39.7	24.2	28.3	29.1	36.7
	Cat-2RF1	25.3	30.1	30.0	**36.9**	37.5	43.3	28.7	32.9	32.1	38.6
	ExHiRD-h	25.3	29.1	28.6	34.7	-	-	28.4	33.5	31.1	37.4
	SEG-Net	21.6	26.5	27.6	36.6	39.6	46.1	28.3	33.2	31.1	37.9
	HyANS	30.0	33.0	33.9	36.1	40.2	46.5	33.0	34.7	33.9	38.9
	HyANS$_h$	**30.8***	**34.3**	**34.6***	**36.9**	**40.7***	**47.2**	**33.2**	**35.5**	**34.5***	**39.5**

The size of hyperbolic word embedding is set as $m = 100$ and the size of hyperbolic hidden state is set as $n = 150$. The word embeddings are initialized first using normal distribution by the method [13], where the gain weight is set as $\sqrt{2}$. Then the embedding is projected into the Poincaré ball by $\exp_0(\cdot)$. In addition, d_b and φ are set as 1.0 and 230 in formula (5) and (6), respectively.

In the training process, we set the batch size as 32. The initial learning rate is set as 0.0008. Early stopping is used when training. In the testing process, our models trained by One2One paradigm use the beam search with a width of 120 and a max depth of 6. Finally our models trained by One2Seq paradigm employ a beam width of 40 and a max depth of 40.

5.5 Results and Analysis

Present keyphrase prediction The results of predicting *present* keyphrases are shown in Table 1. The results show that the generation methods substantially outperform the traditional extraction methods across all the datasets. Among the generation methods, the One2Seq methods can generally achieve better performance than other One2One methods. This improvement may be driven by the inter-relation among keyphrases of each document, which can be effectively captured by the deep models trained by the One2Seq paradigm. In all methods, HyANS$_h$ achieves the best results in term of all metrics on all datasets.

Table 2. Results of predicting *absent* keyphrases of different methods on five datasets.

	Method	Inspec		Krapivin		NUS		SemEval		KP20k	
		F_1@5	R@50	F_1@5	R@50	F_1@5	R@50	F_1@5	R@50	F_1@5	R@50
O2O	CopyRNN$^+$	0.1	8.3	0.9	8.1	1.1	8.1	1.0	2.6	0.8	8.7
	HyAN	**0.3**	8.5	1.1	8.5	1.3	8.5	1.2	2.8	1.2	8.9
	HyAN$_h$	**0.3**	**8.6**	**1.4**	**9.0**	**1.5**	**8.7**	**1.6**	**3.1**	**1.3**	**9.1**
		F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$	F_1@5	F_1@$\mathcal{M}$
One2Seq	CatSeq	0.4	0.8	1.8	3.6	1.6	2.8	1.6	2.8	1.5	3.2
	Cat-2RF1	1.2	2.1	3.0	5.3	1.9	3.1	2.1	3.0	2.7	5.0
	ExHiRD-h	1.1	2.2	2.2	4.3	-	-	1.7	2.5	1.6	3.2
	SEG-Net	1.5	0.9	3.6	1.8	3.6	2.1	**3.0***	2.1	3.6	1.8
	HyANS	1.8	2.7	3.9	6.1	4.0	4.8	2.5	2.9	3.9	4.2
	HyANS$_h$	**2.3***	**3.1**	**4.3***	**7.1**	**5.2***	**5.1**	**3.0***	**3.3**	**4.6***	**5.3**

In all the deep models whether trained by the One2One or One2Seq paradigm, the proposed hyperbolic models outperform the corresponding Euclidean baselines across all the datasets. It should be noted that HyAN can be regarded as CopyRNN and HyANS as CatSeq in hyperbolic space, and they do not use any side information or multi-task learning to achieve better performance like almost all extensions of CoypRNN in Euclidean space, so it is only fair to compare HyAN with CopyRNN, and HyANS with CatSeq. The results show that HyAN and HyANS outperform CopyRNN and CatSeq on all datasets, respectively. This demonstrates the superiority of the hyperbolic methods in modeling hierarchical structures for keyphrase prediction.

Absent keyphrase prediction Unlike present keyphrases, absent keyphrases do not appear in the target document, and thus predicting them is very challenging and requires understanding the latent document semantic. The results are presented in Table 2 (where O2O represents One2One paradigm), where recall R@50 is more suitable for evaluating the performance of One2One methods in absent keyphrase prediction (more detailed descriptions are shown in the work [23]). The results indicate no matter which type of model is trained by One2One or One2Seq paradigm, the proposed hyperbolic models can predict absent keyphrases more accurately than the corresponding Euclidean baselines.

Variable-number keyphrase generation The One2Seq methods can predict a varying number of keyphrases conditioned on the given document, which is one key advantage of this type of method. We conduct experiments on the KP20k dataset to compare the performance of models for generating a varying number of keyphrases in term of both $F_1@\mathcal{O}$ and $F_1@\mathcal{M}$. The results are presented in Table 3. As the results show, HyANS and HyANS$_h$ substantially outperform CatSeq, and HyANS$_h$ achieves the best results in terms of two performance metrics. This indicates that our proposed hierarchy-aware attention mechanism

Table 3. Results of the variable-number keyphrase generation on kp20k dataset.

KP20k	$F_1@\mathcal{O}$	$F_1@\mathcal{M}$
CatSeq	24.3	25.1
HyANS	31.0	32.5
HyANS$_h$	**31.5**	**32.8**

used in HyANS$_h$ is more effective than the primitive hyperbolic attention mechanism [15] used in HyANS.

5.6 Coverage Evaluation of Predicted Keyphrases

As mentioned in the works [21,39], the predicted keyphrases should cover all the main topics discussed in the target document. However, it is challenging to evaluate the degree to which the predicted keyphrases cover the main topics of a target document. To this end, we try to find the ground-truth (*i.e.*, author-provided) keyphrases that are not covered semantically by the predicted keyphrases and use the number of them to measure the *semantic coverage* of predicted keyphrases for a target document.

Specifically, let $G = \{g_i\}_{i=1}^{n}$ be a set of ground-truth keyphrases of a target document, and $K = \{p_j\}_{j=1}^{m}$ be its corresponding set of predicted keyphrases. The number of un-covered ground-truth keyphrases (*uck*) is defined as

$$uck = \sum_{i=1}^{n} \mathbf{1}\Big(\sum_{j=1}^{m} \mathbf{1}\big(s_{ij} = \max_{k=1:n}\{s_{kj}\}\big) = 0 \Big), \tag{11}$$

where s_{ij} is the cosine similarity between embeddings of ground-truth keyphrase g_i and predicted keyphrase p_j, produced by the pre-trained BERT[2]. The indicator function $\mathbf{1}(\cdot)$ outputs 1 if the expression evaluates to true and outputs 0 otherwise. This formula is used to count the number of ground-truth keyphrases, each of which has lower similarities with all the predicted keyphrases. A smaller the value of *uck* suggests a better predictor.

As the results shown in Table 4, our hyperbolic deep models indeed outperform the other two Euclidean models, and HyANS$_h$ gets the best result. The results indicate the predicted keyphrases generated by our hyperbolic models can better cover the topics discussed in a target document and reduce duplicated keyphrases generation.

Table 4. Average number of uncovered author-provided keyphrases (*i.e.*, average *uck*) of different methods on KP20k dataset.

CopyRNN$^+$	HyAN	HyAN$_h$	CatSeq	Cat-2RF1	ExHiRD-h	HyANS	HyANS$_h$
1.7784	1.7602	**1.7385**	1.7729	1.7653	1.7542	1.7334	**1.7328**

[2] `https://github.com/duanzhihua/pytorch-pretrained-BERT`

Table 5. Two examples of generated keyphrases by different methods with the One2Seq training paradigm. Author-assigned (*i.e.*, Gold) keyphrases are shown in bold, and absent keyphrases are labeled by *.

Example 1 **Title**: Active Learning for Software Defect Prediction (#4445 in KP20k) **Abstract**: An **active learning** method, called Two-stage Active learning algorithm (TAL), is developed for software **defect prediction**. Combining the clustering and **support vector machine** techniques, this method improves the performance of the predictor with less labeling effort. Experiments validate its effectiveness. **Gold**: **machine learning***; **defect prediction**; **active learning**; **support vector machine** *HyANS$_h$: **machine learning***; **active learning**; **support vector machine**; support vector machines* Catseq: **active learning**; software defect prediction; clustering; **support vector machine**; software defect prediction Cat-2RF1: **active learning**; software defect prediction; clustering; **support vector machine**; software metrics ExHiRD-h: **active learning**; software defect prediction; clustering; **support vector machine**
Example 2 **Title**: Experience with performance testing of software systems issues, an approach, and case study (#4086 in KP20k) **Gold**: **performance testing**; **software performance testing**; **program testing***; **software testing*** HyANS$_h$: **performance testing**; **software performance testing**; **software testing*** Catseq: **performance testing**; **software performance testing**; **software testing*** Cat-2RF1: **performance testing**; software systems; case study; **software testing*** ExHiRD-h: **performance testing**; software systems; case study; **software testing***; performance evaluation

5.7 Case Study and Visualization

Here, we select two anecdotal examples of research papers shown in Table 5. The predictions generated by different methods along with human-picked "gold" keyphrases are listed in this table. The first paper (*i.e.*, Example 1) presented an active learning method for software defect prediction and assigned "machine learning" as a *absent* keyphrase, which appears in the first position of the author-assigned keyphrase sequence. Obviously, the keyphrase "machine learning" can be regarded as the root topic description of various machine learning methods, such as active learning discussed in this example. As can be seen from Table 5, our hyperbolic HyANS$_h$ is capable of understanding the underlying semantic hierarchical structures in this document, and thus can accurately generate this absent root keyphrase while all the baselines in Euclidean space fail to generate

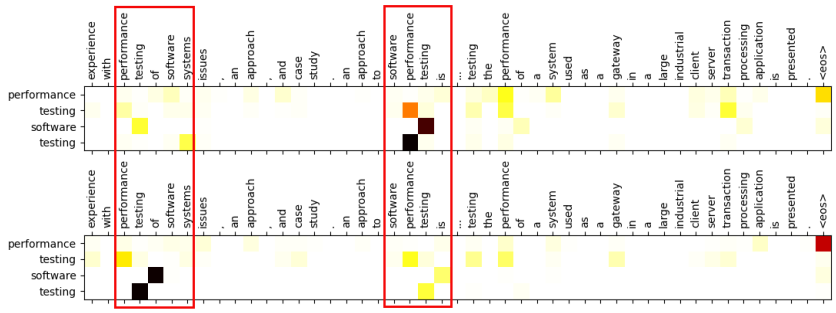

Fig. 3. Attention visualization of hyperbolic HyANS$_h$ (top) and Euclidean Catseq (bottom) on the second example. Deeper shading denotes higher value.

it. This example further indicates that hyperbolic space may help to gain better performance in keyphrase generation.

The second paper (*i.e.*, Example 2) proposed an approach to software performance testing. Comparing with the baseline methods, HyANS$_h$ and Catseq achieve best performance and generate the same keyphrases on this example. Fig.3 visualizes the proposed hyperbolic hierarchy-aware attention in HyANS$_h$ and the Euclidean attention in Catseq to further clarify how our model works. Due to space limitation, we only visualize the first present keyphrase "performance testing" and the absent keyphrase "software testing" in the author-assigned keyphrase sequence, and they are already enough to support our analysis. Although these two keyphreses are correctly generated by both HyANS$_h$ and Catseq, from the results shown in Fig.3, we can clearly see that HyANS$_h$ pays more attention to relevant content words such as "performance" and "testing" while Catseq, to a certain extent, focuses on some irrelevant or functional words such as "is" and "of". This example indicates that compared with the Euclidean space, the hyperbolic space is very helpful for generating keyphrases.

6 Conclusion

In this study, we presented a new solution that aims to predict keyphrases using *hyperbolic* encoder-decoder framework, which can effectively capture the underlying semantic hierarchical structures discussed in a target document. To the best of our knowledge, this is the first study to explore a hyperbolic deep network for keyphrase generation. In addition, we propose a novel hierarchy-aware attention mechanism to further enhance the ability to capture the semantic hierarchical information, and a new metric to measure the degree to which the predicted keyphrases cover the main topics of a target document. Comprehensive experimental results show the proposed hyperbolic models outperform the state-of-the-art Euclidean models across all five datasets. In future, we plan to evaluate the proposed hyperbolic seq2seq model on a large corpus with comprehensive coverage of diverse topics.

Acknowledgements

This work was partially supported by grants from the Scientific Research Project of Tianjin Educational Committee (Grant No. 2021ZD002).

References

1. Ahmad, W.U., Bai, X., Lee, S., Chang, K.W.: Select, extract and generate: Neural keyphrase generation with layer-wise coverage attention. In: Proceedings of ACL (2021)
2. Birman, G.S., Ungar, A.A.: The hyperbolic derivative in the poincaré ball model of hyperbolic geometry. Journal of mathematical analysis and applications **254**(1), 321–333 (2001)
3. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: going beyond euclidean data. IEEE Signal Process. Mag. **34**(4), 18–42 (2017)
4. Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., Jatowt, A.: Yake! keyword extraction from single documents using multiple local features. Information Sciences **509**, 257–289 (2020)
5. Chan, H.P., Chen, W., Wang, L., King, I.: Neural keyphrase generation via reinforcement learning with adaptive rewards. In: Proceedings of ACL (2019)
6. Chen, J., Zhang, X., Wu, Y., Yan, Z., Li, Z.: Keyphrase generation with correlation constraints. In: Proceedings of EMNLP (2018)
7. Chen, W., Chan, H.P., Li, P., Bing, L., King, I.: An integrated approach for keyphrase generation via exploring the power of retrieval and extraction. In: Proceedings of NAACL (2019)
8. Chen, W., Chan, H.P., Li, P., King, I.: Exclusive hierarchical decoding for deep keyphrase generation. In: Proceedings of ACL (2020)
9. Chen, W., Gao, Y., Zhang, J., King, I., Lyu, M.R.: Title-guided encoding for keyphrase generation. In: Proceedings of AAAI (2019)
10. Dhingra, B., Shallue, C.J., Norouzi, M., Dai, A.M., Dahl, G.E.: Embedding text in hyperbolic spaces. In: Proceedings of Twelfth Workshop on TextGraphs (2018)
11. Florescu, C., Caragea, C.: Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In: Proceedings of ACL (2017)
12. Ganea, O., Bécigneul, G., Hofmann, T.: Hyperbolic neural networks. In: Proceedings of NIPS (2018)
13. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of AISTATS (2010)
14. Gu, J., Lu, Z., Li, H., Li, V.O.: Incorporating copying mechanism in sequence-to-sequence learning. In: Proceedings of ACL (2016)
15. Gulcehre, C., Denil, M., Malinowski, M., Razavi, A., Pascanu, R., Hermann, K.M., Battaglia, P., Bapst, V., Raposo, D., Santoro, A., et al.: Hyperbolic attention networks. In: Proceedings of ICLR (2019)
16. Hasan, K.S., Ng, V.: Automatic keyphrase extraction: A survey of the state of the art. In: Proceedings of ACL (2014)
17. Hulth, A., Megyesi, B.B.: A study on automatically extracted keywords in text categorization. In: Proceedings of ACL (2006)
18. Kim, S.N., Medelyan, O., Kan, M.Y., Baldwin, T.: Semeval-2010 task: Automatic keyphrase extraction from scientific articles. In: Proceedings of Workshop on SemEval (2010)

19. Krapivin, M., Autaeu, A., Marchese, M.: Large dataset for keyphrases extraction. Tech. rep., University of Trento (2009)
20. Le, M., Roller, S., Papaxanthos, L., Kiela, D., Nickel, M.: Inferring concept hierarchies from text corpora via hyperbolic embeddings. In: Proceedings of ACL (2019)
21. Liu, Z., Huang, W., Zheng, Y., Sun, M.: Automatic keyphrase extraction via topic decomposition. In: Proceedings of EMNLP (2010)
22. Medelyan, O., Frank, E., Witten, I.H.: Human-competitive tagging using automatic keyphrase extraction. In: Proceedings of EMNLP (2009)
23. Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., Chi, Y.: Deep keyphrase generation. In: Proceedings of ACL (2017)
24. Mihalcea, R., Tarau, P.: Textrank: Bringing order into text. In: Proceedings of EMNLP (2004)
25. Nguyen, T.D., Kan, M.Y.: Keyphrase extraction in scientific publications. In: Proceedings of ICADL (2007)
26. Nickel, M., Kiela, D.: Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In: Proceedings of ICML (2018)
27. Nickel, M., Kiela, D.: Poincaré embeddings for learning hierarchical representations. In: Proceedings of NIPS (2017)
28. Patel, K., Caragea, C.: Exploiting position and contextual word embeddings for keyphrase extraction from scientific papers. In: Proceedings of ECACL (2021)
29. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Proceedings of NIPS (2014)
30. Tang, Y., Huang, W., Liu, Q., Zhang, B.: Qalink: Enriching text documents with relevant Q&A site contents. In: Proceedings of CIKM (2017)
31. Tifrea, A., Bécigneul, G., Ganea, O.E.: Poincaré glove: Hyperbolic word embeddings. In: Proceedings of ICLR (2019)
32. Wan, X., Xiao, J.: Single document keyphrase extraction using neighborhood knowledge. In: Proceedings of AAAI (2008)
33. Wang, L., Cardie, C.: Domain-independent abstract generation for focused meeting summarization. In: Proceedings of ACL (2013)
34. Wang, Y., Li, J., Chan, H.P., King, I., Lyu, M.R., Shi, S.: Topic-aware neural keyphrase generation for social media language. In: Proceedings of ACL (2019)
35. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevillmanning, C.G.: Kea: Practical automatic keyphrase extraction. In: Proceedings of JCDL (1999)
36. Ye, H., Wang, L.: Semi-supervised learning for neural keyphrase generation. In: Proceedings of EMNLP. Proceedings of ACL (2018)
37. Ye, J., Gui, T., Luo, Y., Xu, Y., Zhang, Q.: One2set: Generating diverse keyphrases as a set. In: Proceedings of ACL (2021)
38. Yuan, X., Wang, T., Meng, R., Thaker, K., Brusilovsky, P., He, D.: One size does not fit all: Generating and evaluating variable number of keyphrases. In: Proceedings of ACL (2020)
39. Zhang, Y., Chang, Y., Liu, X., Gollapalli, S.D., Li, X., Xiao, C.: Mike: keyphrase extraction by integrating multidimensional information. In: Proceedings of CIKM (2017)
40. Zhang, Y., Jiang, T., Yang, T., Li, X., Wang, S.: HTKG: Deep keyphrase generation with neural hierarchical topic guidance. In: Proceedings of SIGIR (2022)
41. Zhao, J., Zhang, Y.: Incorporating linguistic constraints into keyphrase generation. In: Proceedings of ACL (2019)