Hyperbolic Relevance Matching for Neural Keyphrase Extraction

Mingyang Song, Yi Feng and Liping Jing[*]
Beijing Key Lab of Traffic Data Analysis and Mining
Beijing Jiaotong University, China
mingyang.song@bjtu.edu.cn

Abstract

Keyphrase extraction is a fundamental task in natural language processing and information retrieval that aims to extract a set of phrases with important information from a source document. Identifying important keyphrase is the central component of the keyphrase extraction task, and its main challenge is how to represent information comprehensively and discriminate importance accurately. In this paper, to address these issues, we design a new hyperbolic matching model (HyperMatch) to represent phrases and documents in the same hyperbolic space and explicitly estimate the phrase-document relevance via the Poincaré distance as the important score of each phrase. Specifically, to capture the hierarchical syntactic and semantic structure information, HyperMatch takes advantage of the hidden representations in multiple layers of RoBERTa and integrates them as the word embeddings via an adaptive mixing layer. Meanwhile, considering the hierarchical structure hidden in the document, HyperMatch embeds both phrases and documents in the same hyperbolic space via a hyperbolic phrase encoder and a hyperbolic document encoder. This strategy can further enhance the estimation of phrase-document relevance due to the good properties of hyperbolic space. In this setting, the keyphrase extraction can be taken as a matching problem and effectively implemented by minimizing a hyperbolic margin-based triplet loss. Extensive experiments are conducted on six benchmarks and demonstrate that HyperMatch outperforms the state-of-the-art baselines.

1 Introduction

Keyphrase Extraction (KE) aims to extract phrases related to the main points discussed in the source document, a fundamental task in Natural Language Processing (NLP). Because of their succinct and accurate expression, keyphrase extraction is helpful for a variety of applications such as information

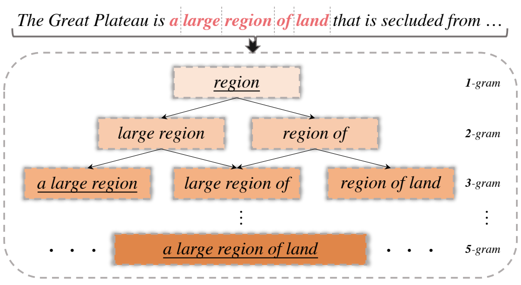

Figure 1: Sample partial of the document in *OpenKP* dataset. For ease of presentation, we assume "a large region of land" is a 5-gram phrase as an example.

retrieval (Kim et al., 2013) and text summarization (Liu et al., 2009a). Typically, keyphrase extraction methods consist of two main components: candidate keyphrase extraction and keyphrase importance estimation. Concretely, the former extracts the candidate keyphrases from the source document via some heuristics (i.e., n-grams are shown in Figure 1), and the latter determines which candidates are chosen as keyphrases. In other words, keyphrase importance estimation directly affects the performance of the keyphrase extraction model in most cases.

Generally, in the neural supervised keyphrase extraction model, keyphrase importance estimation can be subdivided into information representation and importance discrimination. Specifically, information representation focuses on modeling the encoding procedure, and the importance discrimination focuses on measuring and ranking the importance of candidate phrases. To represent information comprehensively, recent methods have been proposed to learn better representations via different backbones, such as Bi-LSTM (Meng et al., 2017), GCNs (Sun et al., 2019; Zhang et al., 2020), and pre-trained language models (e.g., ELMo (Xiong et al., 2019) and BERT (Liu et al., 2020; Sun et al., 2020)). To distinguish the importance of candidate phrases precisely, most existing

[*]Corresponding author.

supervised models (Sun et al., 2020; Mu et al., 2020; Song et al., 2021) estimate and rank the importance of candidate phrases to extract keyphrases by using different approaches, such as classification and ranking models.

Although the methods mentioned above have achieved significant performance, the keyphrase extraction task still needs improvement. Among them, there are the following two main issues. The first issue lies in the information representation. Typically, phrases often exhibit inherent hierarchical structure ingrained with complex syntactic and semantic information (Dai et al., 2020; Alleman et al., 2021; Zhou et al., 2020). In general, the longer phrases contain more complex structures. (as shown in Figure 1, the phrase "a large region of land" has more complex inherent structures than "region" or "a large region". Similarly, the phrase "a large region" is more complex than "region"). Besides the phrases, since linguistic ontologies are intrinsic hierarchies (Dai et al., 2020), the conceptual relations between phrases and the document can also form hierarchical structures. Therefore, the hierarchical structures need to be considered when representing both phrases and documents and estimating the phrase-document relevance. However, it is difficult to capture such structural information even with infinite dimensions in the Euclidean space (Linial et al., 1995). The second issue lies in distinguishing the importance of phrases. Keyphrases are typically used to retrieve and index their corresponding document, so they should be highly related to the main points of the source document (Hasan and Ng, 2014). However, most existing supervised keyphrase extraction methods ignore explicitly modeling the relevance between phrases and their corresponding document, resulting in biased keyphrase extraction.

Motivated by the above issues, we explore the potential of hyperbolic space for the keyphrase extraction task and propose a new hyperbolic relevance matching model (HyperMatch) for neural supervised keyphrase extraction. Firstly, to capture hierarchical syntactic and semantic structure information, HyperMatch integrates the hidden representations in all the intermediate layers of RoBERTa to collect the adaptive contextualized word embeddings via an adaptive mixing layer based on the self-attention mechanism. And then, considering the hierarchical structure hidden in the natural language content, HyperMatch encodes both phrases and documents in the same hyperbolic space via hyperbolic phrase encoder and hyperbolic document encoder. Meanwhile, we adopt the Poincaré distance to calculate the phrase-document relevance by considering the latent hierarchical structures between phrases and the document. In this setting, the keyphrase extraction can be regarded as a matching problem and effectively implemented by minimizing a hyperbolic margin-based triplet loss. To the best of our knowledge, we are the first work to explore the supervised keyphrase extraction in the Hyperbolic space. Extensive experiments on six benchmark datasets show the effectiveness of HyperMatch. The results have demonstrated that HyperMatch outperforms the state-of-the-art baselines in most cases.

2 Preliminaries

Hyperbolic space is an important concept in hyperbolic geometry, which is considered as a special case in the Riemannian geometry (Hopper and Andrews, 2011). Before presenting our model, this section briefly introduces the basic information of hyperbolic space.

In a traditional sense, hyperbolic spaces are not vector spaces; one cannot use standard operations such as summation, multiplication, etc. To remedy this problem, one can utilize the formalism of Möbius gyrovector spaces allowing the generalization of many standard operations to hyperbolic spaces (Khrulkov et al., 2020). Similarly to the previous work (Nickel and Kiela, 2017; Ganea et al., 2018; Tifrea et al., 2019), we adopt the Poincaré ball and use an additional hyper-parameter c which modifies the curvature of Poincaré ball; it is then defined as $\mathbb{D}_c^n = \{\mathbf{x} \in \mathbb{R}^n : c\|\mathbf{x}\|^2 < 1, c \geq 0\}$. The corresponding conformal factor now takes the form $\lambda_{\mathbf{x}}^c := \frac{2}{1-c\|\mathbf{x}\|^2}$. In practice, the choice of c allows one to balance hyperbolic and Euclidean geometries, which is made precise by noting that when $c \to 0$, all the formulas discussed below take their usual Euclidean form.

We restate the definitions of fundamental mathematical operations for the generalized Poincaré ball model. We refer readers to (Ganea et al., 2018) for more details. Next, we give details of the closed-form formulas of several Möbius operations.

Möbius Addition. For a pair $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^n$, the Möbius addition is defined as,

$$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1 + 2c\langle \mathbf{x}, \mathbf{y} \rangle + c\|\mathbf{y}\|^2)\mathbf{x} + (1 - c\|\mathbf{x}\|^2)\mathbf{y}}{1 + 2c\langle \mathbf{x}, \mathbf{y} \rangle + c^2\|\mathbf{x}\|^2\|\mathbf{y}\|^2}. \quad (1)$$

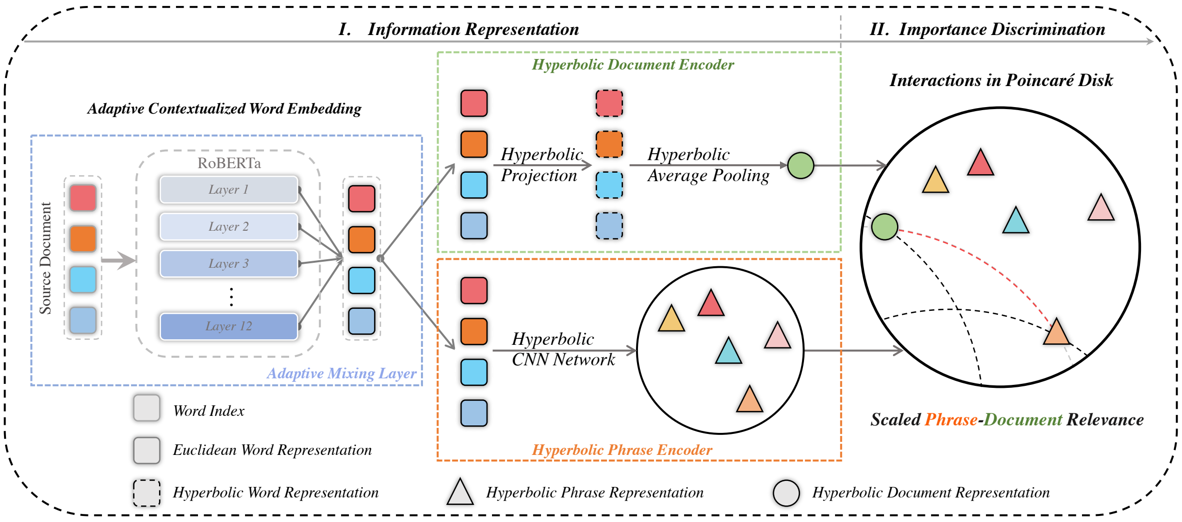
I. Information Representation
II. Importance Discrimination
Adaptive Contextualized Word Embedding
Source Document
RoBERTa
Layer 1
Layer 2
Layer 3
Layer 12
Adaptive Mixing Layer
Hyperbolic Document Encoder
Hyperbolic Projection
Hyperbolic Average Pooling
Hyperbolic CNN Network
Hyperbolic Phrase Encoder
Interactions in Poincaré Disk
Scaled Phrase-Document Relevance
Word Index
Euclidean Word Representation
Hyperbolic Word Representation
Hyperbolic Phrase Representation
Hyperbolic Document Representation

Figure 2: Framework of the hyperbolic relevance matching model (HyperMatch).

Möbius Matrix-vector Multiplication. For a linear map $\mathbf{M}: \mathbb{R}^n \to \mathbb{R}^m$ and $\forall \mathbf{x} \in \mathbb{D}_c^n$, if $\mathbf{M}\mathbf{x} \neq 0$, then the Möbius matrix-vector multiplication is defined as,

$$\mathbf{M} \otimes_c \mathbf{x} = (\frac{1}{\sqrt{c}})\tanh(\frac{\|\mathbf{M}\mathbf{x}\|}{\|\mathbf{x}\|}\tanh^{-1}(\|\sqrt{c}\mathbf{x}\|))\frac{\mathbf{M}\mathbf{x}}{\|\mathbf{M}\mathbf{x}\|}, \tag{2}$$

where $\mathbf{M} \otimes_c \mathbf{x} = 0$ if $\mathbf{M}\mathbf{x} = 0$.

Poincaré Distance. The induced distance function is defined as,

$$d_c(\mathbf{x}, \mathbf{y}) = \frac{2}{\sqrt{c}}\text{arctanh}(\sqrt{c}\|-\mathbf{x} \oplus_c \mathbf{y}\|). \tag{3}$$

Note that with $c = 1$ one recovers the geodesic distance, while with $c \to 0$ we obtain the Euclidean distance $\lim_{c\to 0} d_c(\mathbf{x}, \mathbf{y}) = 2\|\mathbf{x} - \mathbf{y}\|$.

Exponential and Logarithmic Maps. To perform operations in the hyperbolic space, one first needs to define a mapping function from $\mathbb{R}^n$ to $\mathbb{D}_c^n$ to map Euclidean vectors to the hyperbolic space. Let $T_\mathbf{x}\mathbb{D}_c^n$ denote the tangent space of $\mathbb{D}_c^n$ at $\mathbf{x}$. The exponential map $\exp_\mathbf{x}^c(\cdot): T_\mathbf{x}\mathbb{D}_c^n \to \mathbb{D}_c^n$ for $\mathbf{v} \neq 0$ is defined as:

$$\exp_\mathbf{x}^c(\mathbf{v}) = \mathbf{x} \oplus_c (\tanh(\sqrt{c}\frac{\lambda_\mathbf{x}^c\|\mathbf{v}\|}{2})\frac{\mathbf{v}}{\sqrt{c}\|\mathbf{v}\|}). \tag{4}$$

As the inverse of $\exp_\mathbf{x}^c(\cdot)$, the logarithmic map $\log_\mathbf{x}^c(\cdot): \mathbb{D}_c^n \to T_x\mathbb{D}_c^n$ for $\mathbf{y} \neq \mathbf{x}$ is defined as:

$$\log_\mathbf{x}^c(\mathbf{y}) = \frac{2}{\sqrt{c}\lambda_\mathbf{x}^c}\tanh^{-1}(\sqrt{c}\|-\mathbf{x}\oplus_c\mathbf{y}\|)\frac{-\mathbf{x} \oplus_c \mathbf{y}}{\|-\mathbf{x} \oplus_c \mathbf{y}\|} \tag{5}$$

Hyperbolic Averaging Pooling. The average pooling, as an important operation common in natural language processing, is averaging of feature vectors. In the *Euclidean* setting, this operation takes the following form:

$$\text{AP}(\mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_M) = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i. \tag{6}$$

Extension of this operation to hyperbolic spaces is called the *Einstein midpoint* and takes the most simple form in *Klein* coordinates:

$$\text{HyperAP}(\mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_M) = \sum_{i=1}^{M}\gamma_i\mathbf{x}_i / \sum_{i=1}^{M}\gamma_i, \tag{7}$$

where $\gamma_i = \frac{1}{\sqrt{1 - c\|\mathbf{x_i}\|^2}}$ is the *Lorentz* factor. Recent work (Khrulkov et al., 2020) demonstrates that the *Klein* model is supported on the same space as the *Poincaré ball*; however, the same point has different coordinate representations in these models. Let $\mathbf{x}_\mathbb{D}$ and $\mathbf{x}_\mathbb{K}$ denote the coordinates of the same point in the *Poincaré* and *Klein* models correspondingly. Then the following transition formulas hold.

$$\mathbf{x}_\mathbb{D} = \frac{\mathbf{x}_\mathbb{K}}{1 + \sqrt{1 - c\|\mathbf{x}_\mathbb{K}\|^2}}, \tag{8}$$

$$\mathbf{x}_\mathbb{K} = \frac{2\mathbf{x}_\mathbb{D}}{1 + c\|\mathbf{x}_\mathbb{D}\|^2}. \tag{9}$$

Therefore, given points in the *Poincaré ball*, we can first map them to the *Klein* model via Eq.(9), compute the average using Eq.(7), and then move it back to the *Poincaré* model via Eq.(8).

3 HyperMatch

Given a document $\mathcal{D} = \{w_1, ..., w_i, ..., w_M\}$, the candidate phrases are first extracted from the source document by the n-gram structures, where M indicates the max length of the input document. Then, to determine which candidates are keyphrases, we design a new hyperbolic relevance matching model (HyperMatch), which consists of two main procedures: information representation and importance discrimination. Figure 2 illustrates the overall framework of HyperMatch.

3.1 Information Representation

Information representation is one of the essential parts of keyphrase importance estimation, which needs to represent information comprehensively. To capture rich syntactic and semantic information, HyperMatch first embeds words by the pre-trained language model RoBERTa with the adaptive mixing layer. Then, phrases and documents are embedded in the same hyperbolic space by the hyperbolic phrase encoder and hyperbolic document encoder. In the following subsections, the information representation procedure will be described in detail.

3.1.1 Contextualized Word Encoder

Pre-trained language models (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019) have emerged as a critical technology for achieving impressive gains in natural language tasks. These models extend the idea of word embeddings by learning contextualized text representations from large-scale corpora using a language modeling objective. Thus, recent keyphrase extraction methods (Xiong et al., 2019; Sun et al., 2020; Wang et al., 2020; Mu et al., 2020) represent words / documents by the last intermediate layer of pre-trained language models.

However, various probing tasks (Jawahar et al., 2019; de Vries et al., 2020) are proposed to discover linguistic properties learned in contextualized word embeddings, which demonstrates that different intermediate layers in pre-trained language models contain different linguistic properties or information. Specifically, each layer has specific specializations, so combining features from different layers may be more beneficial than selecting the last one based on the best overall performance.

Motivated by the phenomenon above, we propose a new adaptive mixing layer to combine all intermediate layers of RoBERTa (Liu et al., 2019) to obtain representations. Firstly, each word in the source document $\mathcal{D}$ is represented by all intermediate layers in RoBERTa, which is encoded to a sequence of vector $\mathbf{H} = \{\mathbf{h}_1, ..., \mathbf{h}_i, ..., \mathbf{h}_M\}$,

$$\mathbf{H} = \text{RoBERTa}\{\mathbf{w}_1, ..., \mathbf{w}_i, ..., \mathbf{w}_M\}. \qquad (10)$$

Specially, $\mathbf{h}_i \in \mathbb{R}^{L*d_r}$ indicates the i-th contextualized word embedding of $\mathbf{w}_i$, where L and d_r are set to 12 and 768. Then, the self-attention mechanism is adopted to aggregate multi-layer representations of each word as follows:

$$\alpha_i = \text{softmax}(\mathbf{V}_a \mathbf{h}_i), \qquad (11)$$

$$\hat{\mathbf{h}}_i = \mathbf{W}_a \alpha_i \mathbf{h}_i, \qquad (12)$$

where $\mathbf{V}_a \in \mathbb{R}^{d_r}$ and $\mathbf{W}_a \in \mathbb{R}^{d_r * d_r}$ are learnable weights. Here, $\alpha_i \in \mathbb{R}^L$ represents the adaptive mixing weights of the proposed adaptive mixing layer. In this case, each word in the source document $\mathcal{D}$ is transferred to a sequence of vector $\hat{\mathbf{H}} = \{\hat{\mathbf{h}}_1, ..., \hat{\mathbf{h}}_i, ..., \hat{\mathbf{h}}_M\}$. The adaptive mixing layer allows our model to obtain more comprehensive word embeddings, capturing more meaningful information (e.g., surface, syntactic, and semantic).

3.1.2 Hyperbolic Phrase Encoder

Phrases often exhibit inherent hierarchies ingrained with complex syntactic and semantic information (Zhu et al., 2020). Therefore, representing information requires sufficiently encoding semantic and syntactic information, especially for the latent hierarchical structures hidden in the natural languages. Recent studies (Sun et al., 2020; Xiong et al., 2019) typically obtain phrase representations in Euclidean space, which makes it difficult to learn representations with such latent structural information even with infinite dimensions in Euclidean space (Linial et al., 1995). On the contrary, hyperbolic spaces are non-Euclidean geometric spaces that can naturally capture the latent hierarchical structures (Sarkar, 2011; Sa et al., 2018).

Lately, the use of the hyperbolic space in NLP (Dhingra et al., 2018; Tifrea et al., 2019; Nickel and Kiela, 2017) is motivated by the ubiquity of hierarchies (e.g., the latent hierarchical structures in phrases, sentences, and documents) in NLP tasks. Therefore, in this paper, we propose to embed phrases in the hyperbolic space. Concretely, the phrase representation of the i-th n-gram c_i^n is computed as follows,

$$\hat{\mathbf{h}}_i^n = \text{CNN}^n(\hat{\mathbf{h}}_{i:i+n}), \qquad (13)$$

where $\hat{\mathbf{h}}_i^n \in \mathbb{R}^{d_h}$ represents the i-th n-gram representation, $n \in [1, N]$ indicates the length of n-grams, and N is the maximum length of n-grams.

Each n-gram has its own set of convolution filters CNN^n with window size n and stride 1.

To capture the latent hierarchies of phrases, we map phrases representation to the Poincaré ball using the *exponential* map,

$$\tilde{\mathbf{h}}_i^n = \exp_{\mathbf{0}}^c(\hat{\mathbf{h}}_i^n), \quad (14)$$

where $\tilde{\mathbf{h}}_i^n$ indicates the i-th n-gram $\hat{\mathbf{h}}_i^n$ phrase representation in the hyperbolic space. By mapping phrases representation into hyperbolic spaces, HyperMatch may implicitly model the latent hierarchical structure of phrases.

3.1.3 Hyperbolic Document Encoder

When using the source document as the query to match keyphrases, the document representation should cover its main points (important information). Meanwhile, documents are usually long text sequences with richer semantic and syntactic information than phrases. Many current BERT-based methods (Mu et al., 2020; Zhong et al., 2020) in NLP obtain documents representation by using the first output token (the [CLS] token) of pre-trained language models.

However, recent studies (Reimers and Gurevych, 2019; Li et al., 2020) demonstrate that in many NLP tasks, documents representation obtained by the average pooling of words representation is better than the [CLS] token. Motivated by the above methods, we use the average pooling, a simple and effective operation, to encode documents. To further consider the latent hierarchical structures of documents, we map word representations and transfer the average pooling operation to the hyperbolic space. In this case, we first map word representations to the hyperbolic space via the *exponential* map as follows:

$$\tilde{\mathbf{H}} = \{\tilde{\mathbf{h}}_1, ..., \tilde{\mathbf{h}}_i, ..., \tilde{\mathbf{h}}_M\} = \exp_{\mathbf{0}}^c(\hat{\mathbf{H}}\mathbf{W}_h), \quad (15)$$

where $\mathbf{W}_h \in \mathbb{R}^{d_r * d_h}$ maps the original BERT embedding space to the tangent space of the origin of the Poincaré ball. Then $\exp_{\mathbf{0}}(\cdot)$ maps the tangent space inside the Poincaré ball. Next, we use the hyperbolic averaging pooling to encode the source document as follows:

$$\tilde{\mathbf{h}} = \text{HyperAP}(\{\tilde{\mathbf{h}}_1, ..., \tilde{\mathbf{h}}_i, ..., \tilde{\mathbf{h}}_M\}), \quad (16)$$

where $\tilde{\mathbf{h}} \in \mathbb{R}^{d_h}$ indicates the hyperbolic document representation (called the *Einstein midpoint* pooling vectors in the Poincaré ball (Gulcehre et al., 2019)). The hyperbolic average pooling emphasizes semantically specific words that usually contain more information but occur less frequently than general ones. It should be noted that points near the boundary of the Poincaré ball get larger weights in the *Einstein midpoint* formula, which are regarded to be more representative content (more helpful information such as the latent hierarchies) from the source document (Dhingra et al., 2018; Zhu et al., 2020).

3.2 Importance Discrimination

Importance discrimination is one of the primary parts of keyphrase importance estimation, which measures and sorts the importance of candidate phrases accurately to extract keyphrases. To reach this goal, we first calculate the scaled phrase-document relevance between phrases and their corresponding document via the Poincaré distance as the important score of each candidate phrase. Then, the important score is optimized by the margin-based triplet loss to extract keyphrases.

3.2.1 Scaled Phrase-Document Relevance

Besides the intrinsic hierarchies of linguistic ontologies, the conceptual relations between candidate phrases and their corresponding document can also form hierarchical structures. Once the document representation $\tilde{\mathbf{h}}$ and phrase representations $\tilde{\mathbf{h}}_i^n$ are obtained, it is expected that the phrases and their corresponding document embedded close to each other based on their geodesic distance[1] if they are highly relevant. Specifically, the scaled phrase-document relevance of the i-th n-gram representation c_i^n can be computed as follows:

$$S(c_i^n, \mathcal{D}) = -\frac{\lambda(d_c(\tilde{\mathbf{h}}_i^n, \tilde{\mathbf{h}}))^2}{\sqrt{d_h}} + (1-\lambda) f_c(\tilde{\mathbf{h}}_i^n), \quad (17)$$

where $S(\cdot)$ indicates the scaled phrase-document relevance. Here, d_c indicates the Poincaré distance, which is introduced in Eq.(3). Furthermore, f_c indicates the linear transformation in the hyperbolic space. Specifically, for Eq. 17, the first term indicates modeling the phrase-document relevance explicitly, and the second term denotes modeling the phrase-document relevance implicitly. Estimating the phrase-document relevance via the Poincaré distance in hyperbolic space allows HyperMatch to

[1]Note that cosine similarity (Wang et al., 2017) is not appropriate to be the metric since there does not exist a clear hyperbolic inner-product for the *Poincaré ball* (Tifrea et al., 2019), so the Poincaré distance is more suitable.

model the relationships between candidate phrases and their document by simultaneously considering semantics and latent hierarchical structures, which benefits ranking keyphrases accurately. Furthermore, we find that with increasing representation dimension d_h, the value of the phrase-document relevance will also increase, resulting in model optimization crash, and the loss value tends to infinity. To counteract this effect, we scale the phrase-document relevance by $\frac{1}{\sqrt{d_h}}$.

3.2.2 Margin-based Triplet Loss

To select phrases with higher importances, we adopt the margin-based triplet loss in our model and optimize for margin separation in the hyperbolic space. Therefore, we first treat the candidate keyphrases in the document that are labelled as keyphrases, in the positive set $\mathbf{P}^+$, and the others to the negative set $\mathbf{P}^-$, to obtain the matching labels. Then, the loss function is calculated as follows:

$$\mathcal{L} = \max(0, \frac{\delta}{\sqrt{d_h}} - S(p^+, \mathcal{D}) + S(p^-, \mathcal{D})), \quad (18)$$

where δ indicates the margin. It enforces HyperMatch to sort the candidate keyphrases p^+ ahead of p^- within their corresponding document. Through this training objective, our model will tend to extract the keyphrases, which are more relevant to the source document.

4 Experimental Settings

4.1 Benchmark Datasets

Six benchmark keyphrase datasets are used in our experiments, which contain *OpenKP* (Xiong et al., 2019), *KP20k* (Meng et al., 2017), *Inspec* (Hulth, 2003), *Krapivin* (Krapivin and Marchese, 2009), *Nus* (Nguyen and Kan, 2007), and *SemEval* (Kim et al., 2010)). We follow the previous work (Sun et al., 2020) to preprocess each dataset with the same method.

4.2 Implementation Details

Implementation details of HyperMatch are summarized in Table 1. The maximum document length is 512 tokens due to RoBERTa limitations (Liu et al., 2019) and documents are zero-padded or truncated to this length. Our model was implemented in Pytorch 1.8[2] (Paszke et al., 2019) using the huggingface reimplementation of RoBERTa[3] (Wolf et al., 2019) and was trained on eight NVIDIA RTX A4000 GPUs to achieve best performance.

[2]https://pytorch.org/

[3]https://huggingface.co/transformers/index.html

Hyperparameter	Dimension or Value
RoBERTa Embedding ($\mathbb{R}^{d_c}$)	768
Hyperbolic Rank ($\mathbb{R}^{d_h}$)	768
Max Sequence Length	512
Maximum Phrase Length (N)	5
c	1
λ	0.5
δ	1.0
Optimizer	AdamW
Batch Size	72
Learning Rate	5×10^{-5}
Warm-Up Proportion	10%

Table 1: Parameters used for training HyperMatch.

4.3 Evaluation Metrics

For the keyphrase extraction task, the model's performance is typically evaluated by comparing the top-K predicted keyphrases with the target keyphrases (ground-truth labels). The evaluation cutoff K can be a fixed number (e.g., F1@5 compares the top-5 keyphrases predicted by the model with the ground-truth to compute an F1 score). Following the previous work (Meng et al., 2017; Sun et al., 2019), we adopt macro-averaged recall and F-measure (F1) as evaluation metrics, and K is set to be 1, 3, 5, and 10. In the evaluation, we apply Porter Stemmer[4] (Porter, 2006) to both target keyphrases and extracted keyphrases when determining the exact match of keyphrases.

4.4 Baselines

We compare two kinds of solid baselines to give a comprehensive evaluation of the performance of HyperMatch: unsupervised keyphrase extraction models (e.g., TextRank (Mihalcea and Tarau, 2004) and TFIDF (Jones, 2004)) and supervised keyphrase extraction models (e.g., classification and ranking models based variants of BERT (Sun et al., 2020)). Noticeably, HyperMatch extracts keyphrases without using additional features on the *OpenKP* dataset. Therefore, for the sake of fairness, we do not compare with the methods (Xiong et al., 2019; Wang et al., 2020) which use additional features to extract keyphrases. In addition, this paper mainly focuses on exploring keyphrase extraction in hyperbolic space via a matching framework (sim-

[4]https://tartarus.org/martin/PorterStemmer/

Model	*OpenKP*								
	P@1	P@3	P@5	R@1	R@3	R@5	F1@1	**F1@3**	F1@5
Unsupervised Keyphrase Extraction Models									
TFIDF	28.3	18.4	13.7	15.0	28.4	34.7	19.6*	22.3*	19.6*
TextRank	7.7	6.2	5.5	4.1	9.8	14.2	5.4*	7.6*	7.9*
Supervised Keyphrase Extraction via Classification Models									
BERT-Spanning-KPE	47.6	28.5	20.9	25.3	43.6	52.1	31.8	33.2	28.9
BERT-Chunking-KPE	51.1	30.6	22.5	27.1	46.4	55.8	34.0	35.6	31.1
SpanBERT-Chunking-KPE	52.3	32.1	23.5	27.8	48.6	58.1	34.8	37.2	32.4
RoBERTa-Chunking-KPE	53.3	32.2	23.5	28.3	48.6	58.1	35.5	37.3	32.4
Supervised Keyphrase Extraction via Ranking Models									
BERT-Ranking-KPE	51.3	32.3	23.5	27.3	48.9	58.2	34.2	37.4	32.5
SpanBERT-Ranking-KPE	53.0	32.7	24.0	28.4	49.7	59.3	35.5	38.0	33.1
RoBERTa-Ranking-KPE	53.8	33.7	24.4	29.0	50.9	60.4	36.1	39.0	33.7
HyperMatch	**54.7**	**33.9**	**24.7**	**29.5**	**51.5**	**61.2**	**36.4**	**39.4**	**33.8**

Table 2: Model performance on the *OpenKP* dataset. The best results of our model are highlighted in bold. F1@3 is the main evaluation metric (marked in bold) for this dataset (Xiong et al., 2019; Wang et al., 2020). * denotes these results are not included in the original paper and are estimated with Precision and Recall score. The results of the baselines are reported in their corresponding papers.

ilar to the ranking model). Hence, the compared baselines we mainly choose are keyphrase extraction methods based on the classification and ranking models rather than some existing studies based on integration models (Chen et al., 2019; Ahmad et al., 2021; Wu et al., 2021) or multi-task learning (Song et al., 2021).

5 Results and Analysis

In this section, we investigate the performance of HyperMatch on six widely-used benchmark keyphrase extraction datasets (OpenKP, KP20k, Inspec, Krapivin, Nus, and Semeval) from three facets. The first one demonstrates its superiority by comparing HyperMatch with the recent baselines in terms of several metrics. The second one is to verify the effect of each component via ablation tests. The last one is to analyze the sensitivity of the triplet loss with different margins.

5.1 Performance Comparison

The experimental results are given in Table 2 and Table 3. Overall, HyperMatch outperforms the recent BERT-based keyphrase extraction models (the results are reported in their own articles) in most cases. Concretely, on the *OpenKP* and *KP20k* datasets, HyperMatch achieves better results than the best ranking models RoBERTa-Ranking-KPE. We consider that the main reason for this result may be that learning representation in hyperbolic space can capture more latent hierarchical structures than the Euclidean space. Meanwhile, compared with the results on the other four zero-shot datasets (*Inspec*, *Krapivin*, *Nus*, and *Semeval*) in Table 3, it can be seen that HyperMatch outperforms both unsupervised and supervised baselines. We consider that the main reason is the scaled phrase-document relevance *explicitly* models a strong connection between phrases and their corresponding document via the Poincaré distance, obtaining more robust performance even in different datasets.

5.2 Ablation Study

In this section, we report on several ablation experiments to analyze the effect of different components. The ablation experiment on the *OpenKP* dataset is shown in Table 4.

To measure the effectiveness of the hyperbolic space for the keyphrase extraction task, we compare it with the same model in the Euclidean space and use the Euclidean distance to explicitly model the phrase-document relevance. As shown in Table 4, HyperMatch outperforms EuclideanMatch, which shows that the hyperbolic space can capture the latent hierarchical structures more effectively than the Euclidean space.

To verify the effectiveness of the adaptive mixing layer, we propose a model HyperMatch w/o

Model	***Inspec***		***Krapivin***		***Nus***		***SemEval***		***KP20k***	
	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10
TFIDF	22.3	30.4	11.3	14.3	13.9	18.1	12.0	18.4	10.5	13.0
TextRank	22.9	27.5	17.2	14.7	19.5	19.0	17.2	18.1	18.0	15.0
RoBERTa-Ranking-KPE†	28.1	29.1	29.9	23.7	44.6	37.7	35.4	32.6	41.4	34.2
HyperMatch	**30.4**	**32.2**	**32.8**	**26.3**	**45.8**	**41.3**	**35.7**	**36.8**	**41.6**	**34.3**

Table 3: Results of keyphrase extraction on five benchmark keyphrase datasets. F1 scores on the top 5 and 10 keyphrases are reported. † indicates that these results are evaluated via the code which is provided by its corresponding paper. The best results are highlighted in bold.

Model	***OpenKP***		
	F1@1	**F1@3**	F1@5
HyperMatch	**36.4**	**39.4**	**33.7**
EuclideanMatch	36.1	38.5	33.4
HyperMatch w/o Relevance	36.1	38.9	33.6
HyperMatch w/o AML	36.3	38.7	33.5

Table 4: Ablation tests on the *OpenKP* dataset. The best results are highlighted in bold. F1@3 is the main evaluation metric (marked in bold) for this dataset.

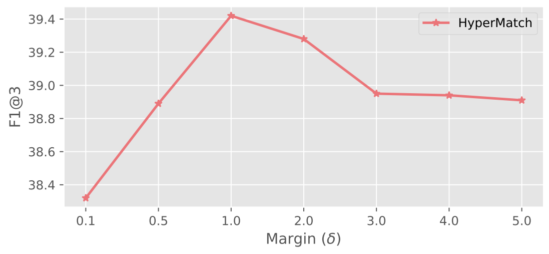

Figure 3: Performance of HyperMatch with different margins (δ) of the margin-based triplet loss on the *OpenKP* dataset.

AML, which indicates HyperMatch without using the adaptive mixing layer module and only uses the last intermediate layer of RoBERTa to embed phrases and documents. As shown in Table 4, the performance drops in all evaluation metrics without the adaptive mixing layer. These results demonstrate that combining all the intermediate layers of RoBERTa by the self-attention mechanism to embed words can capture more helpful information of different layers in RoBERTa.

Unlike our model, most recent keyphrase extraction methods (e.g., RoBERTa-Ranking-KPE) implicitly model relevance between candidate phrases and their corresponding document by a linear transformation layer as the phrase-document relevance. Therefore, to verify the effectiveness of explicitly modeling the phrase-document relevance, we built the HyperMatch w/o Relevance, which only implicitly computes the phrase-document relevance by a hyperbolic linear transformation layer (Ganea et al., 2018). The results of HyperMatch w/o Relevance show a drop in all evaluation metrics, indicating that explicitly considering the relevance between phrases and the document is essential for estimating the importance of candidate phrases in the keyphrase extraction task.

5.3 Sensitivity of Hyperparameters

In this section, we verify the sensitivity of HyperMatch with different margins (δ) of the hyperbolic triplet loss. For keyphrase extraction methods equipped with the margin-based triplet loss, the margin design significantly impacts the final result, where a poor margin usually causes performance degradation. Therefore, we verify the effects of different margins on HyperMatch in Figure 3. We can see that HyperMatch achieves the best results when $\delta = 1$.

6 Related Work

This section briefly describes the related work from two fields: keyphrase extraction and hyperbolic deep learning.

6.1 Keyphrase Extraction

Most existing KE models are based on the two-stage extraction framework, which consists of two main parts: candidate keyphrase extraction and keyphrase importance estimation. Candidate keyphrase extraction extracts a set of candidate phrases from the source document by heuristics (e.g., essential n-gram-based phrases (Hulth, 2004; Medelyan et al., 2009; Xiong et al., 2019; Sun et al., 2020; Wang et al., 2020)). Keyphrase importance estimation first represents candidate phrases and documents by the pre-trained language models (Devlin et al., 2019; Liu et al., 2019) and then estimates the phrase-document relevance implicitly as the importance scores. Finally, the candidate phrases are ranked by their importance scores, which can be learned by either unsupervised (Mihalcea and Tarau, 2004; Liu et al., 2009b) or supervised (Xiong et al., 2019; Sun et al., 2020; Mu et al., 2020) ranking approaches.

Different from the existing KE models, we first embed phrases and documents by RoBERTa in the Euclidean space and then map these representations to the same hyperbolic space to capture the latent hierarchical structures. Next, we adopt the Poincaré distance to model the phrase-document relevance explicitly as the important score of each candidate phrase. Finally, the hyperbolic margin-based triplet loss is used to optimize the whole model for extracting keyphrases. To the best of our knowledge, we are the first study to explore supervised keyphrase extraction in the hyperbolic space.

6.2 Hyperbolic Deep Learning

Recent Studies on representation learning (Nickel and Kiela, 2017; Tifrea et al., 2019; Mathieu et al., 2019) demonstrate that the hyperbolic space is more suitable for embedding symbolic data with hierarchies than the Euclidean space since the tree-like properties (Hamann, 2018) of the hyperbolic space make it efficient to learn hierarchical representations with low distortion (Sa et al., 2018; Sarkar, 2011). As linguistic ontologies are innately hierarchies, hierarchies are ubiquitous in natural language (Dai et al., 2020). Some recent studies show the superiority of the hyperbolic space for many natural language processing tasks (Gulcehre et al., 2019; Zhu et al., 2020). Chen et al. (2021) demonstrate that mapping contextualized word embeddings (i.e., BERT-based embeddings) to hyperbolic space can capture richer hierarchical structure information than the euclidean space when encoding natural language text.

7 Conclusions and Future Work

A new hyperbolic relevance matching model HyperMatch is proposed to map phrase and document representations into the hyperbolic space and model the relevance between candidate phrases and the document via the Poincaré distance. Specifically, HyperMatch first combines the intermediate layers of RoBERTa via the adaptive mixing layer for capturing richer syntactic and semantic information. Then, phrases and documents are encoded in the same hyperbolic space to capture the latent hierarchical structures. Next, the phrase-document relevance is estimated explicitly via the Poincaré distance as the importance scores of all the candidate phrases. Finally, we adopt the hyperbolic margin-based triplet loss to optimize the whole model for extracting keyphrases.

In this paper, we explore the hyperbolic space to implicitly model the latent hierarchical structures when representing candidate phrases and documents. In the future, it will be interesting to introduce external knowledge (i.e., WordNet) to explicitly model the latent hierarchical structures when representing candidate phrases and documents. Our code is publicly available to facilitate other research[5].

8 Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0106800; the National Science Foundation of China under Grant 61822601 and 61773050; the Beijing Natural Science Foundation under Grant Z180006; the Fundamental Research Funds for the Central Universities (2019JBZ110).

References

Wasi Ahmad, Xiao Bai, Soomin Lee, and Kai-Wei Chang. 2021. Select, extract and generate: Neural keyphrase generation with layer-wise coverage attention. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing.*, pages 1389–1404.

Matteo Alleman, Jonathan Mamou, Miguel A Del Rio, Hanlin Tang, Yoon Kim, and SueYeon Chung. 2021. Syntactic perturbations reveal representational correlates of hierarchical phrase structure in pretrained language models.

Boli Chen, Yao Fu, Guangwei Xu, Pengjun Xie, Chuanqi Tan, Mosha Chen, and Liping Jing. 2021. Probing bert in hyperbolic spaces. In *International Conference on Learning Representations.*

[5]https://github.com/MySong7NLPer/HyperMatch

Wang Chen, Hou Pong Chan, Piji Li, Lidong Bing, and Irwin King. 2019. An integrated approach for keyphrase generation via exploring the power of retrieval and extraction. In *NAACL-HLT (1)*, pages 2846–2856. Association for Computational Linguistics.

Shuyang Dai, Zhe Gan, Yu Cheng, Chenyang Tao, Lawrence Carin, and Jingjing Liu. 2020. Apo-vae: Text generation in hyperbolic space. *CoRR*, abs/2005.00054.

Wietse de Vries, Andreas van Cranenburgh, and Malvina Nissim. 2020. What's so special about bert's layers? a closer look at the nlp pipeline in monolingual and multilingual models. In *EMNLP (Findings)*, pages 4339–4350. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186. Association for Computational Linguistics.

Bhuwan Dhingra, Christopher J. Shallue, Mohammad Norouzi, Andrew M. Dai, and George E. Dahl. 2018. Embedding text in hyperbolic spaces. In *TextGraphs@NAACL-HLT*, pages 59–69. Association for Computational Linguistics.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *NeurIPS*, pages 5350–5360.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter W. Battaglia, Victor Bapst, David Raposo, Adam Santoro, and Nando de Freitas. 2019. Hyperbolic attention networks. In *ICLR (Poster)*. OpenReview.net.

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

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *ACL (1)*, pages 1262–1273. The Association for Computer Linguistics.

C. Hopper and B. Andrews. 2011. *The Ricci Flow in Riemannian Geometry*.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *EMNLP*.

Anette Hulth. 2004. Enhancing linguistically oriented automatic keyword extraction. In *HLT-NAACL (Short Papers)*. The Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL (1)*, pages 3651–3657. Association for Computational Linguistics.

Karen Spärck Jones. 2004. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation*, 60(5):493–502.

Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan V. Oseledets, and Victor S. Lempitsky. 2020. Hyperbolic image embeddings. In *CVPR*, pages 6417–6427. IEEE.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. Semeval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *SemEval@ACL*, pages 21–26. The Association for Computer Linguistics.

Youngsam Kim, Munhyong Kim, Andrew Cattle, Julia Otmakhova, Suzi Park, and Hyopil Shin. 2013. Applying graph-based keyword extraction to document retrieval. In *IJCNLP*, pages 864–868. Asian Federation of Natural Language Processing / ACL.

M. Krapivin and M. Marchese. 2009. Large dataset for keyphrase extraction.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *EMNLP (1)*, pages 9119–9130. Association for Computational Linguistics.

Nathan Linial, Eran London, and Yuri Rabinovich. 1995. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245.

Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. 2009a. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *HLT-NAACL*, pages 620–628. The Association for Computational Linguistics.

Rui Liu, Zheng Lin, and Weiping Wang. 2020. Keyphrase prediction with pre-trained language model.

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

Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009b. Clustering to find exemplar terms for keyphrase extraction. In *EMNLP*, pages 257–266. ACL.

Emile Mathieu, Charline Le Lan, Chris J. Maddison, Ryota Tomioka, and Yee Whye Teh. 2019. Continuous hierarchical representations with poincaré variational auto-encoders. In *NeurIPS*, pages 12544–12555.

O. Medelyan, E. Frank, and I. H. Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Internat. Conference of Empirical Methods in Natural Language Processing, EMNLP-2009,*.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *ACL*, pages 582–592. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *EMNLP*, pages 404–411. ACL.

Funan Mu, Zhenting Yu, Lifeng Wang, Yequan Wang, Qingyu Yin, Yibo Sun, Liqun Liu, Teng Ma, Jing Tang, and Xing Zhou. 2020. Keyphrase extraction with span-based feature representations. *CoRR*, abs/2002.05407.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *ICADL*, volume 4822 of *Lecture Notes in Computer Science*, pages 317–326.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *NIPS*, pages 6338–6347.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237. Association for Computational Linguistics.

M.F. Porter. 2006. An algorithm for suffix stripping. *Program: Electronic Library and Information Systems*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. Cite arxiv:1908.10084Comment: Published at EMNLP 2019.

Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. 2018. Representation tradeoffs for hyperbolic embeddings. *CoRR*, abs/1804.03329.

Rik Sarkar. 2011. Low distortion delaunay embedding of trees in hyperbolic plane. In *Graph Drawing*, volume 7034 of *Lecture Notes in Computer Science*, pages 355–366.

Mingyang Song, Liping Jing, and Lin Xiao. 2021. Importance Estimation from Multiple Perspectives for Keyphrase Extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Si Sun, Chenyan Xiong, Zhenghao Liu, Zhiyuan Liu, and Jie Bao. 2020. Joint keyphrase chunking and salience ranking with bert. *CoRR*, abs/2004.13639.

Zhiqing Sun, Jian Tang, Pan Du, Zhi-Hong Deng, and Jian-Yun Nie. 2019. Divgraphpointer: A graph pointer network for extracting diverse keyphrases. In *SIGIR*, pages 755–764.

Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea. 2019. Poincare glove: Hyperbolic word embeddings. In *ICLR (Poster)*. OpenReview.net.

Yansen Wang, Zhen Fan, and Carolyn Penstein Rosé. 2020. Incorporating multimodal information in open-domain web keyphrase extraction. In *EMNLP (1)*, pages 1790–1800. Association for Computational Linguistics.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *IJCAI*, pages 4144–4150.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Huanqin Wu, Wei Liu, Lei Li, Dan Nie, Tao Chen, Feng Zhang, and Di Wang. 2021. Unikeyphrase: A unified extraction and generation framework for keyphrase prediction. *CoRR*, abs/2106.04847.

Lee Xiong, Chuan Hu, Chenyan Xiong, Daniel Campos, and Arnold Overwijk. 2019. Open domain web keyphrase extraction beyond language modeling. In *EMNLP/IJCNLP (1)*, pages 5174–5183. Association for Computational Linguistics.

Haoyu Zhang, Dingkun Long, Guangwei Xu, Pengjun Xie, Fei Huang, and Ji Wang. 2020. Keyphrase extraction with dynamic graph convolutional networks and diversified inference. *CoRR*, abs/2010.12828.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *ACL*, pages 6197–6208. Association for Computational Linguistics.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020. Parsing all: Syntax and semantics, dependencies and spans. In *EMNLP (Findings)*, pages 4438–4449. Association for Computational Linguistics.

Yudong Zhu, Di Zhou, Jinghui Xiao, Xin Jiang, Xiao Chen, and Qun Liu. 2020. Hypertext: Endowing fasttext with hyperbolic geometry. In *EMNLP (Findings)*, pages 1166–1171. Association for Computational Linguistics.