A Fully Hyperbolic Neural Model for Hierarchical Multi-Class Classification

Federico López **Michael Strube**
Heidelberg Institute for Theoretical Studies
Research Training Group AIPHES
firstname.lastname@h-its.org

Abstract

Label inventories for fine-grained entity typing have grown in size and complexity. Nonetheless, they exhibit a hierarchical structure. Hyperbolic spaces offer a mathematically appealing approach for learning hierarchical representations of symbolic data. However, it is not clear how to integrate hyperbolic components into downstream tasks. This is the first work that proposes a fully hyperbolic model for multi-class multi-label classification, which performs all operations in hyperbolic space. We evaluate the proposed model on two challenging datasets and compare to different baselines that operate under Euclidean assumptions. Our hyperbolic model infers the latent hierarchy from the class distribution, captures implicit hyponymic relations in the inventory, and shows performance on par with state-of-the-art methods on fine-grained classification with remarkable reduction of the parameter size. A thorough analysis sheds light on the impact of each component in the final prediction and showcases its ease of integration with Euclidean layers. [1]

1 Introduction

Entity typing classifies textual mentions of entities, according to their semantic class, within a set of labels (or classes) organized in an inventory. The task has progressed from recognizing a few *coarse* classes (Sang and De Meulder, 2003), to extremely large inventories, with hundreds (Gillick et al., 2014) or thousands of labels (Choi et al., 2018). Therefore, exploiting inter-label correlations has become critical to improve performance.

Large inventories tend to exhibit a hierarchical structure, either by an explicit tree-like arrangement of the labels (*coarse* labels at the top, *fine-grained* at the bottom), or implicitly through the label distribution in the dataset (*coarse* labels appear more frequently than *fine-grained* ones). Prior work has integrated only explicit hierarchical information by formulating a hierarchy-aware loss (Murty et al., 2018; Xu and Barbosa, 2018) or by representing instances and labels in a joint Euclidean embedding space (Shimaoka et al., 2017; Abhishek et al., 2017). However, the resulting space is hard to interpret, and these methods fail to capture implicit relations in the label inventory. Hyperbolic space is naturally equipped for embedding symbolic data with hierarchical structures (Nickel and Kiela, 2017). Intuitively, that is because the amount of space grows exponentially as points move away from the origin. This mirrors the exponential growth of the number of nodes in trees with increasing distance from the root (Cho et al., 2019) (see Figure 1).

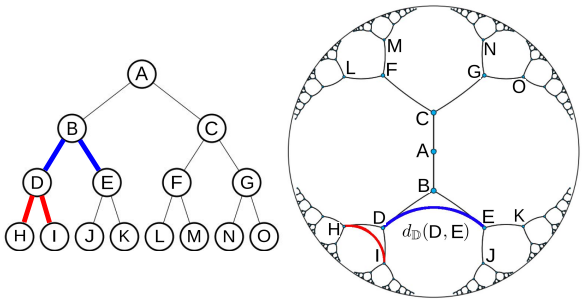


Figure 1: Tree embedded in hyperbolic space. Items at the top of the hierarchy are placed near the origin of the space, and lower items near the boundary. Moreover, the hyperbolic distance (Eq. 1) between sibling nodes resembles the one through the common ancestor, analogous to the distance in the tree. That is $d_{\mathbb{D}}(\mathrm{D},\mathrm{E}) \approx d_{\mathbb{D}}(\mathrm{D},\mathrm{B}) + d_{\mathbb{D}}(\mathrm{B},\mathrm{E})$.

In this work, we propose a fully hyperbolic neural model for fine-grained entity typing. Noticing a perfect match between hierarchical label inventories in the linguistic task and the benefits of hyperbolic spaces, we endow a classification model with

[1] Code available at: https://github.com/nlpAThits/hyfi

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 460–475
November 16 - 20, 2020.

a suitable geometry to capture this fundamental property of the data distribution. By virtue of the hyperbolic representations, the proposed approach automatically infers the latent hierarchy arising from the class distribution and achieves a meaningful and interpretable organization of the label space. This arrangement captures implicit hyponymic relations (*is-a*) in the inventory and enables the model to excel at fine-grained classification. To the best of our knowledge, this work is the first to apply hyperbolic geometry from beginning to end to perform multi-label classification on real NLP datasets.

Recent work has proposed hyperbolic neural components, such as word embeddings (Tifrea et al., 2019), recurrent neural networks (Ganea et al., 2018) and attention layers (Gulcehre et al., 2019). However, researchers have incorporated these isolated components into neural models, whereas the rest of the layers and algorithms operate under Euclidean assumptions. This impedes models from fully exploiting the properties of hyperbolic geometry. Furthermore, there are different analytic models of hyperbolic space, and not all previous work operates in the same one, which hinders their combination, and hampers their adoption for downstream tasks (e.g. Tifrea et al. (2019) learn embeddings in the Poincaré model, Gulcehre et al. (2019) aggregate points in the Klein model, or Nickel and Kiela (2018) perform optimization in the Lorentz model). We address these issues. Our model encodes textual inputs, applies a novel attention mechanism, and performs multi-class multi-label classification, executing all operations in the Poincaré model of hyperbolic space (§4).

We evaluate the model on two datasets, namely Ultra-Fine (Choi et al., 2018) and OntoNotes (Gillick et al., 2014), and compare to Euclidean baselines as well as to state-of-the-art methods for the task (Xiong et al., 2019; Onoe and Durrett, 2019). The hyperbolic system has competitive performance when compared to an ELMo model (Peters et al., 2018) and a BERT model (Devlin et al., 2019) on very fine-grained types, with remarkable reduction of the parameter size (§6). Instead of relying on large pre-trained models, we impose a suitable inductive bias by choosing an adequate metric space to embed the data, which does not introduce extra burden on the parameter footprint.

By means of the exponential and logarithmic maps (explained in §2) we are able to mix hyperbolic and Euclidean components into one model, aiming to exploit their strengths at different levels of the representation. We perform a thorough ablation that allows us to understand the impact of each hyperbolic component in the final performance of the system (§6.1.1 and §6.1.2), and showcases its ease of integration with Euclidean layers.

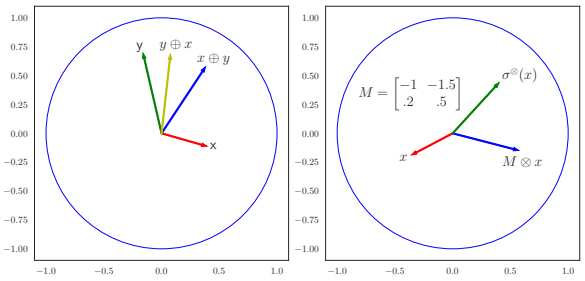


Figure 2: Visualization of Möbius operations. Left: Möbius addition (noncommutative). Right: Matrix-vector multiplication and pointwise non-linearity.

2 Hyperbolic Neural Networks

In this section we briefly recall the necessary background on hyperbolic neural components. The terminology and formulas used throughout this work follow the formalism of Möbius gyrovector spaces (Ungar, 2008a,b), and the definitions of hyperbolic neural components of Ganea et al. (2018). For more information about Riemannian geometry and Möbius operations see Appendix A and B. In the following, $\langle \cdot, \cdot \rangle$ and $\| \cdot \|$ are the inner product and norm inherited from the Euclidean space.

Hyperbolic space: It is a non-Euclidean space with constant negative curvature. We adopt the Poincaré ball model of hyperbolic space (Cannon et al., 1997). In the general n-dimensional case, it becomes $\mathbb{D}^n = \{x \in \mathbb{R}^n \mid \|x\| < 1\}$[2]. The Poincaré model is a *Riemannian manifold* equipped with the *Riemannian metric* $g_x^{\mathbb{D}} = \lambda_x^2 g^E$, where $\lambda_x := \frac{2}{1-\|x\|^2}$ is called the *conformal factor* and $g^E = \mathrm{I}_n$ is the Euclidean metric tensor. The distance between two points $x, y \in \mathbb{D}^n$ is given by:

$$d_{\mathbb{D}}(x, y) = \cosh^{-1}\left(1 + 2\frac{\|x - y\|^2}{(1 - \|x\|^2)(1 - \|y\|^2)}\right) \quad (1)$$

Möbius addition: It is the hyperbolic analogous to vector addition in Euclidean space. Given two

[2]Ganea et al. (2018) define the ball as $\mathbb{D}^n = \{x \in \mathbb{R}^n \mid c\|x\|^2 < 1\}$ with a parameter c in relation to the radius of the Poincaré ball $r = 1/\sqrt{c}$. In this work we assume $c = 1$ therefore we omit such parameter.

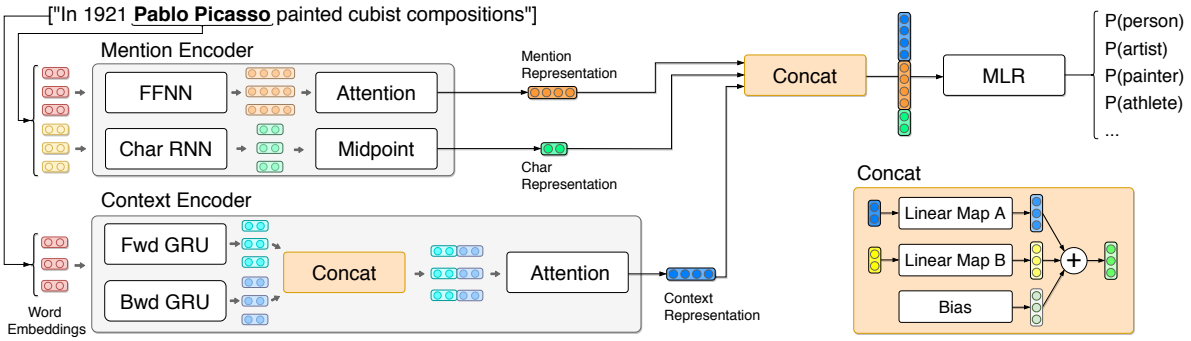


Figure 3: Overview of the proposed model. The mention encoder extracts word and char-level entity representations. The context encoder is based on a bidirectional-GRU with attention. The outputs of both encoders are concatenated and passed to a classifier based on a multinomial logistic regression.

points $x, y \in \mathbb{D}^n$, it is defined as:

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

Möbius matrix-vector multiplication: Given a linear map $M : \mathbb{R}^n \to \mathbb{R}^m$, which we identify with its matrix representation, and a point $x \in \mathbb{D}^n$, $Mx \neq 0$, it is defined as:

$$M \otimes x = \tanh\left(\frac{\|Mx\|}{\|x\|}\tanh^{-1}(\|x\|)\right)\frac{Mx}{\|Mx\|} \quad (3)$$

Pointwise non-linearity: If we model it as $\varphi : \mathbb{R}^n \to \mathbb{R}^n$, then its Möbius version $\varphi^{\otimes}$ can be applied using the same formulation of the matrix-vector multiplication. A visualization of the aforementioned operations can be seen in Figure 2.

By combining these operations we obtain a one-layer feed-forward neural network (FFNN) in hyperbolic space, described as $y = \varphi^{\otimes}(M \otimes x \oplus b)$ with $M \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{D}^m$ as trainable parameters. Note that the parameter b lies in the hyperbolic space, thus its updates during training need to be corrected for this geometry.

Exponential and logarithmic maps: For each point $x \in \mathbb{D}^n$, let $T_x\mathbb{D}^n$ denote the associated *tangent space*, which is always a subset of Euclidean space (Liu et al., 2019). We make use of the exponential map $\exp_x : T_x\mathbb{D}^n \to \mathbb{D}^n$ and the logarithmic map $\log_x : \mathbb{D}^n \to T_x\mathbb{D}^n$ to map points in the hyperbolic space to the Euclidean space, and vice-versa. At the origin of the space, they are given for $v \in T_{\mathbf{0}}\mathbb{D}^n \backslash \{0\}$ and $y \in \mathbb{D}^n \backslash \{0\}$:

$$\begin{aligned} \exp_{\mathbf{0}}(v) &= \tanh(\|v\|)\frac{v}{\|v\|} \\ \log_{\mathbf{0}}(y) &= \operatorname{arctanh}(\|y\|)\frac{y}{\|y\|} \end{aligned} \quad (4)$$

To map a point $y \in \mathbb{D}^n$ onto the Euclidean space we apply $\log_{\mathbf{0}}(y)$. Conversely, to map a point $v \in \mathbb{R}^n$ onto the hyperbolic space, we assume $\mathbb{R}^n = T_{\mathbf{0}}\mathbb{D}^n$ and apply $\exp_{\mathbf{0}}(v)$. This allows to mix hyperbolic and Euclidean neural layers as shown in §6.1.2.

3 Fine-grained Entity Typing

Given a context sentence s containing an entity mention m, the goal of entity typing is to predict the correct type labels t_m that describe m from a type inventory T. The ground-truth type set t_m may contain multiple types, making the task a multi-class multi-label classification problem.

For fine-grained entity typing the type inventory T tends to contain hundreds to thousands of labels. Encoding hierarchical information from large type inventories has been proven critical to improve performance (López et al., 2019). Thus we hypothesize that our proposed hyperbolic model will benefit from this representation.

4 Hyperbolic Classification Model

In this section we propose a general hyperbolic neural model for classification with sequential data as input. The building blocks are defined in a generic manner such that they can be applied to different tasks, or integrated with regular Euclidean layers. Our proposed architecture resembles recent neural models applied to entity typing (Choi et al., 2018). For the encoders we employ the neural networks introduced in Ganea et al. (2018), we propose a novel attention mechanism operating entirely in the Poincaré model, and we extend the hyperbolic classifier to multi-class multi-label setups. An overview of the model can be seen in Figure 3.

4.1 Mention Encoder

To represent the mention, we combine word and char-level features, similar to Lee et al. (2017). Given a sequence of k tokens in a mention span, we represent them using pre-trained word embeddings $w_i \in \mathbb{D}^n$ which we assume to lie in hyperbolic space. We apply a hyperbolic FFNN, described as:

$$m_i = \tanh^{\otimes}(W^M \otimes w_i \oplus b^M) \quad (5)$$

with $m_i \in \mathbb{D}^{d_M}$, and where $W^M \in \mathbb{R}^{d_M \times n}, b^M \in \mathbb{D}^{d_M}$ are parameters of the model. We combine the resulting $m_1, ..., m_k$ into a single mention representation $\mathbf{m} \in \mathbb{D}^{d_M}$ by computing a weighted sum of the token representations in hyperbolic space with the attention mechanism explained in §4.4.

Moreover, we extract features from the sequence of characters in the mention span with a recurrent neural network (RNN) (Lample et al., 2016). We represent each character with a char-embedding $c_i \in \mathbb{D}^{d_C}$ that we train in the Poincaré ball. An RNN operating in hyperbolic space is defined by:

$$h_{t+1} = \varphi^{\otimes}(W^C \otimes h_t \oplus U^C \otimes c_t \oplus b^C) \quad (6)$$

where $W^C, U^C \in \mathbb{R}^{d_C \times d_C}, b^C, h_t \in \mathbb{D}^{d_C}$, and φ is a pointwise non-linearity function. Finally, we obtain a single representation $\mathbf{c} \in \mathbb{D}^{d_C}$ by taking the midpoint of the states h_i using Equation 9.

4.2 Context Encoder

To encode the context we apply a hyperbolic version of gated recurrent units (GRU) (Cho et al., 2014) proposed in Ganea et al. (2018)[3]. Given a sequence of l tokens, we represent them with a pre-trained word embedding $w_i \in \mathbb{D}^n$, and apply a forward and backward GRU, producing contextualized representations $\overrightarrow{h_i}, \overleftarrow{h_i} \in \mathbb{D}^{d_S}$ for each token. We concatenate the resulting states into a single embedding $s_i = \text{concat}(\overrightarrow{h_i}, \overleftarrow{h_i})$ (see concat in §4.3), where $s_i \in \mathbb{D}^{2d_S}$. Ultimately, we combine $s_1, ..., s_l$ into a single context representation $\mathbf{s} \in \mathbb{D}^{2d_S}$ with the distance-based attention mechanism.

4.3 Concatenation

If we model the concatenation of two vectors in the Poincaré ball as appending one to the other, this does not guarantee that the result remains inside the ball. Thus, we apply a generalized version of the concatenation operation. For $x \in \mathbb{D}^k, y \in \mathbb{D}^l$, then $\text{concat} : \mathbb{D}^k \times \mathbb{D}^l \to \mathbb{D}^n$ is defined as:

$$\text{concat}(x, y) = M_1 \otimes x \oplus M_2 \otimes y \oplus b \quad (7)$$

where $M_1 \in \mathbb{R}^{n \times k}, M_2 \in \mathbb{R}^{n \times l}, b \in \mathbb{D}^n$ are parameters of the model. In Euclidean architectures, the concatenation of vectors is usually followed by a linear layer, which takes the form of Equation 7 when written explicitly.

4.4 Distance-based Attention

Previous approaches to hyperbolic attention (Gulcehre et al., 2019; Chami et al., 2019) require mappings of points to different spaces, which hinders their adoption into neural models. We propose a novel attention mechanism in the Poincaré model of hyperbolic space. We cast attention as a weighted sum of vectors in this geometry, without requiring any extra mapping of the inputs. In this manner, we make consistent use of the same analytical model of hyperbolic space across all components, which eases their integration.

To obtain the attention weights, we exploit the hyperbolic distance between points (Gulcehre et al., 2019). Given a sequence of states $x_i \in \mathbb{D}^n$, we combine them with a trainable position embedding $p_i \in \mathbb{D}^n$ such that $r_i = x_i \oplus p_i$. We use addition as the standard method to enrich the states with positional information (Vaswani et al., 2017; Devlin et al., 2019). We apply two different linear transformations on r_i to obtain vectors q_i and k_i, both lying in the Poincaré ball. We compute the distance between these two points and finally obtain the weight by applying a softmax over the sequence in the following manner:

$$\begin{gathered} q_i = W^Q \otimes r_i \oplus b^Q, \quad k_i = W^K \otimes r_i \oplus b^K \\ \alpha(q_i, k_i) = \text{softmax}(-\beta d_{\mathbb{D}}(q_i, k_i)) \end{gathered} \quad (8)$$

where $W^Q, W^K \in \mathbb{R}^{n \times n}, b^Q, b^K \in \mathbb{D}^n$ and $\beta \in \mathbb{R}$ are parameters of the model. Attention weights will be higher for elements with q_i and k_i vectors placed close to each other.

The positional embeddings are trained along with the model as a hyperbolic parameter. For the context encoder, they reflect relative distances between the i-th word and the entity mention. For the mention encoder, they represent the absolute position of the word inside the mention span.

To aggregate the points as a weighted summation in hyperbolic space we propose to apply the

[3]For a complete description of this network see Appendix C or Ganea et al. (2018) §3.3

Möbius midpoint, which obeys many of the properties that we expect from a weighted average in Euclidean space (Ungar (2010), Theorem 4.6):

$$m = \frac{1}{2} \otimes \frac{\sum_{i=1}^{n} \alpha_i \gamma(x_i)^2 x_i}{\sum_{i=1}^{n} \alpha_i \left(\gamma(x_i)^2 - \frac{1}{2}\right)} \tag{9}$$

where x_i are the states in the sequence, α_i the weights corresponding to each state, and $\gamma(x_i)$ the Lorentz factors. By applying the Möbius midpoint we develop an attention mechanism that operates entirely in the Poincaré model of hyperbolic space. Detailed formulas and experimental observations can be found in Appendix D.

4.5 Classification in the Poincaré Ball

The input of the classifier is the concatenation of mention and context features. To perform multiclass classification in the Poincaré ball, we adapt the generalized multinomial logistic regression (MLR) from Ganea et al. (2018). Given K classes and $k \in \{1, ..., K\}$, $p_k \in \mathbb{D}^m$, $a_k \in T_{p_k}\mathbb{D}^m \backslash \{0\}$, the formula for the hyperbolic MLR is:

$$p(y = k|x) \propto f\left(\lambda_{p_k} \|a_k\| \sinh^{-1}\left(\frac{2\langle -p_k \oplus x, a_k\rangle}{(1 - \| - p_k \oplus x\|^2)\|a_k\|}\right)\right) \tag{10}$$

Where $x \in \mathbb{D}^m$, and p_k and a_k are trainable parameters. It is based on formulating logits as distances to margin hyperplanes. The hyperplanes in hyperbolic space are defined by the union of all geodesics passing through p_k and orthogonal to a_k.

Although this formulation was made for one-label classification, the underlying notion also holds for multi-label setups. In that case, we need to be able to select several classes by considering the distances (logits) to all hyperplanes. To achieve that we employ the $\mathrm{sigmoid}$ function as f, instead of a $\mathrm{softmax}$, and predict the given class if $p(y = k|x) > 0.5$. More details in Appendix E.

Figure 4 shows examples of the hyperbolic definition of multiple hyperplanes, which follow the curvature of the space.

4.6 Optimization

With the proposed classification model, we aim to minimize variants of the binary cross-entropy loss function as the training objective.

The model has trainable parameters in both Euclidean and hyperbolic space. We apply the Geoopt implementation of *Riemannian Adam* (Kochurov et al., 2020) as a Riemannian adaptive optimization method (Bécigneul and Ganea, 2019) to carry out a gradient-based update of the parameters in their respective geometry.

5 Experiments

We evaluate the proposed hyperbolic model on two different datasets for fine-grained entity typing, and compare to Euclidean baselines as well as state-of-the-art models.

5.1 Data

For analysis and evaluation of the model, we focus on the Ultra-Fine entity typing dataset (Choi et al., 2018). It contains 10,331 target types defined as free-form noun phrases and divided in three levels of granularity: *coarse*, *fine* and *ultra-fine*. Besides this segregation, the dataset does not provide any further explicit information about the relations among the types. The data consist of 6,000 crowdsourced examples and 6M training samples in the open-source version, automatically extracted with distant supervision. Our evaluation is done on the original crowdsourced dev/test splits.

To gain a better understanding of the proposed model, we also experiment on the OntoNotes dataset (Gillick et al., 2014) as it is a standard benchmark for entity typing.

5.2 Setup

The MLR classifier operates in a hyperbolic space of m dimensions with $m = d_M + d_C + 2d_S$. By setting different values, we experiment with three models: BASE ($m = 100$), LARGE ($m = 250$) and XLARGE ($m = 500$).

As word embeddings we employ Poincaré GloVe embeddings (Tifrea et al., 2019), which are pretrained in the Poincaré model. Hence, the input to the encoders is already in hyperbolic space and all operations can be performed in this geometry. These embeddings are not updated during training. Low values of dropout are used since the model was very sensitive to this parameter given the behaviour of the hyperbolic distance.

On the Ultra-Fine dataset, for each epoch, we train over the entire training set, and we run extra iterations over the crowdsourced split before evaluating. In this way, the model benefits from the large amount of noisy, automatically-generated data, and is fine-tuned with high-quality crowdsourced samples. As previous work (Xiong et al.,

	Total			Coarse			Fine			Ultra-Fine			
Model	**P**	**R**	**F_1**	**P**	**R**	**F_1**	**P**	**R**	**F_1**	**P**	**R**	**F_1**	**# Params**
DENOISED	50.7	33.1	**40.1**	66.9	**80.7**	73.2	41.7	46.2	43.8	45.6	17.4	25.2	31.0M
BERT	**51.6**	32.8	**40.1**	**67.4**	80.6	**73.4**	41.6	**54.7**	**47.3**	**46.3**	15.6	23.4	110.0M
LABELGCN	49.3	28.1	35.8	66.2	68.8	67.5	**43.9**	40.7	42.2	42.4	14.2	21.3	5.1M
MULTITASK	48.0	23.0	31.0	60.0	61.0	61.0	40.0	38.0	39.0	42.0	8.0	14.0	6.1M
HY BASE	48.5	29.1	36.3	64.4	72.2	68.1	39.4	38.5	38.9	39.3	14.5	21.2	1.8M
HY LARGE	42.3	33.5	37.4	63.6	72.1	67.6	36.3	48.3	41.4	33.3	19.7	24.7	4.6M
HY XLARGE	43.4	**34.2**	38.2	61.4	73.9	67.1	35.7	46.6	40.4	36.5	**19.9**	**25.7**	9.5M

Table 1: Macro-averaged P, R and F_1 on the Ultra-Fine dev set for different baselines and models. We only reproduced LABELGCN. Values for other baselines are taken from the original publications.

2019; Onoe and Durrett, 2019), we optimize the multi-task objective proposed by Choi et al. (2018).

For evaluation we report Macro-averaged and Micro-averaged F_1 metrics computed from the precision/recall scores over the same three granularities established by Choi et al. (2018). For all models we optimize *Total* Macro-averaged F_1 on the validation set, and evaluate on the test set. Following Ganea et al. (2018), we report the average of three runs given the highly non-convex spectrum of hyperbolic neural networks. Hyperparameters are detailed in Appendix F along with other practical aspects to ensure numerical stability.

5.3 Baselines

Euclidean baseline: We replace all operations of the hyperbolic model by their Euclidean counterpart. To map the Poincaré GloVe embeddings to the Euclidean space we apply $\log_{\mathbf{0}}$. We do not apply any kind of normalization or correction over the weights to circumscribe them into the unit ball. On the contrary, we grant them freedom over the entire Euclidean space to establish a fair comparison.

Multi-task: Model proposed by Choi et al. (2018), along with the Ultra-Fine dataset.

LabelGCN: Model introduced by Xiong et al. (2019). A label-relational inductive bias is imposed by means of a graph propagation layer that encodes label co-occurrence statistics.

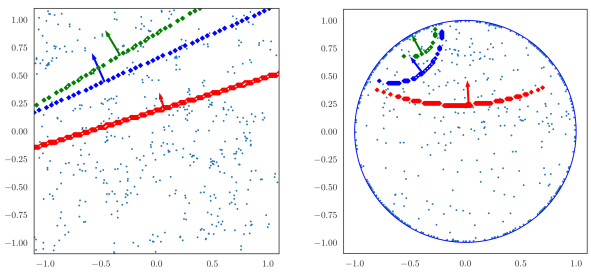
BERT: We compare to the setup of Onoe and Durrett (2019) in which BERT (Devlin et al., 2019) is adapted for this task and fine-tuned on the crowd-sourced train split.

Denoised: An ELMo-based model (Peters et al., 2018) proposed by Onoe and Durrett (2019) trained on raw and denoised distantly-labeled data.

6 Results and Discussion

Following previous work (Choi et al., 2018; Onoe and Durrett, 2019), we report results on the development set in Table 1. All hyperbolic models outperform MULTITASK and LABELGCN baselines on *Total* Macro F_1. DENOISED and BERT systems, based on large pre-trained models, show the best *Total* performance. Nonetheless, HY XLARGE has a competitive performance, and surpasses both systems on *ultra-fine* F_1. In the hyperbolic model, fine-grained types are placed near the boundary of the ball, where the amount of space grows exponentially. Furthermore, the underlying structure of the type inventory is hierarchical, thus the hyperbolic definition of the hyperplanes is well-suited to improve the classification in this case (see comparison with Euclidean classifiers on Figure 4). These properties combined enable the hyperbolic model to excel at classifying hierarchical labels, with outstanding improvements on very fine-grained types.

The reduction of the parameter size is also remarkable: 70% and 91% versus DENOISED and BERT respectively. This emphasizes the importance of choosing a suitable metric space that fits the data distribution (hierarchical in this case) as a powerful and efficient inductive bias. Through adequate tools and formulations, we are able to exploit this bias without introducing an overload



(a) Euclidean Space. (b) Hyperbolic Space.

Figure 4: Classification hyperplanes for the types `person` (red), `artist` (blue) and `musician` (green). The hyperbolic formulation of the hyperplanes is better suited for hierarchical inventories.

person		artist		musician	
Types	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$
artist	0.26	*musician*	0.25	*singer*	0.24
author	0.28	actor	0.26	actor	0.25
actor	0.30	person	0.26	artist	0.25
speaker	0.30	author	0.26	composer	0.27
leader	0.30	singer	0.28	band	0.27

Table 2: Closest p_k points in the Poincaré Ball to different Ultra-Fine entity types. The model is able to capture hierarchical relations such as `singer` *is-a* `musician` *is-a* `artist` *is-a* `person`.

on the parameter cost.

Correspondence of results between HY BASE and LABELGCN suggest that both models capture similar information. LABELGCN requires label co-occurrence statistics represented as a weighted graph, from where a hierarchy can be easily derived (Krioukov et al., 2010). The similarity of results indicates that the hyperbolic model is able to implicitly encode the latent hierarchical information in the label co-occurrences without additional inputs or the burden of the graph layer.

To shed light on this aspect, we inspect the points p_k learned by HY BASE to define the hyperplanes of Equation 10. Table 2 shows the types corresponding to the closest points to the label `person` and its *subtypes*, measured by hyperbolic distance. The types are highly correlated given that they often co-occur in similar contexts. Moreover, the model captures hyponymic relations (*is-a*) present in the label co-occurrences. An analogous behaviour is observed for other types (see tables in Appendix G). The inductive bias given by the hyperbolic geometry allows the model to capture the hierarchy, deriving a meaningful and interpretable representation of the label space: *coarse* labels near the origin, *fine-grained* labels near the boundary, and hyponymic relations are preserved. It is also noteworthy that the model learns these relations automatically without requiring the explicit data encoded in the graph.

6.1 Comparison of the Spaces

A comparison of the metric spaces for different models on the test set is shown in Table 3. It can be seen that the hyperbolic model outperforms its Euclidean variants in all settings. It is notable that this trend holds even in high-dimensional spaces (500 dimensions for XLARGE). Since the label inventory exhibits a clearly hierarchical structure, it perfectly suits the hyperbolic classification method.

		Coarse		Fine		Ultra	
Model		**Ma**	**Mi**	**Ma**	**Mi**	**Ma**	**Mi**
BASE	HY	**69.6**	**67.3**	**42.0**	**39.7**	21.2	19.1
	EU	68.5	66.1	39.8	36.5	17.8	16.1
LARGE	HY	67.9	65.4	38.4	36.3	24.3	22.3
	EU	67.1	63.8	36.7	34.7	22.0	19.7
XLARGE	HY	69.1	66.2	39.7	37.2	**26.1**	**24.0**
	EU	67.9	65.4	37.8	35.3	22.2	20.0

Table 3: Results on Ultra-Fine test set for macro and micro F_1 across metric spaces and dimensions.

The hyperbolic model brings considerable gains as the granularity becomes finer: 5.1% and 16.2% relative improvement in *fine* and *ultra-fine* Macro F_1 respectively for the BASE model over its Euclidean counterpart. We also observe that as the size of the model increases, the Euclidean baseline becomes more competitive for *ultra-fine*. This is due to the Euclidean model gaining enough capacity to accommodate the separation hyperplanes with higher dimensions, thus reducing the gap.

It is noticeable that the BASE model outperforms the larger ones on *coarse* and *fine* granularities. That is due to the larger models overfitting given the low dropout applied. Moreover, Euclidean and hyperbolic models exhibit a similar performance on the *coarse* granularity when compared to each other. A possible explanation is that the separation planes for these labels are located closer to the origin of the space. In this region, the spaces behave alike in terms of the distance calculation, and this similarity is reflected in the results as well.

6.1.1 Word Embeddings Ablation

The input for both the Euclidean and hyperbolic models are Poincaré GloVe embeddings, which are originally trained in hyperbolic space (Tifrea et al., 2019). This might favor the hyperbolic model, despite the application of the $\log_{\mathbf{0}}$ map in the Euclidean case. Thus, we replace the hyperbolic embeddings by the regular GloVe embeddings (Pennington et al., 2014), and use $\exp_{\mathbf{0}}$ on the hyperbolic model to project them into the ball.

Table 4 shows that the tendency of the BASE hyperbolic model outperforming the Euclidean one

BASE	Coarse		Fine		Ultra	
Model	**Ma**	**Mi**	**Ma**	**Mi**	**Ma**	**Mi**
HY GLOVE	**68.7**	**66.6**	**41.5**	**38.8**	**22.1**	**20.1**
EU GLOVE	67.8	65.3	39.7	36.0	20.7	18.6

Table 4: Test results on Ultra-Fine. Poincaré GloVe embeddings are replaced by regular GloVe embeddings.

Model	Coarse		Fine		Ultra	
	Ma	Mi	Ma	Mi	Ma	Mi
HY BASE	**69.6**	**67.3**	**42.0**	**39.7**	21.2	19.1
EU Encoder	68.8	66.3	41.7	38.9	**22.0**	**20.1**
EU Attention	68.9	66.4	40.8	38.0	20.1	18.4
EU Concat	68.6	66.1	40.6	37.5	21.8	19.8
EU MLR	69.2	67.1	40.8	38.0	17.3	15.8

Table 5: Results on Ultra-Fine test set. Ablation of the hyperbolic model, replacing one component by its Euclidean counterpart at a time.

holds, and that the improvement does not come from the embeddings. Also, in this way we showcase how the hyperbolic model can be easily integrated with regular word embeddings.

6.1.2 Component Ablation

With the aim of analyzing the contribution of the different hyperbolic components, we perform an ablation study on the BASE model. We divide the system in *encoder*, *attention* (both in the mention and context encoders), *concatenation*, and *MLR*, and replace them, one at a time, by their Euclidean counterparts. Note that when Euclidean and hyperbolic components are mixed, we convert the internal representations from one manifold to the other with the $\exp_{\mathbf{0}}$ and $\log_{\mathbf{0}}$ maps.

As we see in Table 5, MLR is the component that contributes the most to the *ultra-fine* classification. The hierarchical structure of the type inventory combined with the hyperbolic definition of the hyperplanes are the reason of this (see Figure 4).

Hyperbolic attention and concatenation are relevant for *coarse* and *fine-grained* classification (here is where the biggest drop appears when they are removed), but do not play a major role in the *ultra-fine* granularity.

Finally, the encoders do not benefit from the hyperbolic representation. As the reason for this we consider that the model is not able to capture tree-like relations among the input tokens such that they can be exploited for the task.

This ablation suggests that the main benefits of hyperbolic layers arise when they are incorporated at deeper levels of representation in the model, and not over low-level features or raw text.

Computing time: Möbius operations are more expensive than their Euclidean counterparts. Due to this, in our experiments we found the hyperbolic encoder to be twice slower, and the MLR 1.5 times slower than their Euclidean versions.

Model		Coarse		Fine		Ultra	
		Ma	Mi	Ma	Mi	Ma	Mi
BASE	HY	82.0	80.2	41.8	**41.4**	23.9	25.0
	EU	81.8	80.3	37.7	36.1	17.5	15.8
LARGE	HY	**83.1**	**81.3**	**42.0**	**41.4**	**24.0**	**25.2**
	EU	82.4	80.9	38.2	36.7	18.9	18.1

Table 6: Macro and micro F_1 on OntoNotes test set for different granularities.

6.2 OntoNotes Dataset

To further understand the capabilities of the proposed model we also perform an evaluation on the OntoNotes dataset (Gillick et al., 2014). In this case, we apply the standard binary cross-entropy loss, since fine-grained labels are scarce in this dataset. Following previous work (Xiong et al., 2019), we train over the dataset augmented by Choi et al. (2018). Results for the three granularities for BASE and LARGE models are presented in Table 6. The hyperbolic models outperform the Euclidean baselines in both cases, and the difference is noticeable for *fine* and *ultra-fine* (42.0 vs 38.2 and 24.0 vs 18.9 on Macro F_1 for the LARGE model), in accordance with the results on Ultra-Fine.

We report a comparison with neural systems in Table 7. The hyperbolic model, without requiring the explicit hierarchy provided in this dataset, achieves a competitive performance. Nonetheless, the advantages of the hyperbolic model are mitigated by the low multiplicity of fine-grained labels, and the lower hierarchy.

7 Related Work

Type inventories for the task of fine-grained entity typing (Ling and Weld, 2012; Yosef et al., 2012) have grown in size and complexity (Del Corro et al., 2015; Choi et al., 2018). Researchers have tried to incorporate hierarchical information on the type distribution in different manners (Shimaoka et al., 2016; Ren et al., 2016a). Shimaoka et al. (2017) encode the hierarchy through a sparse matrix. Xu

Model	Acc.	Ma-F_1	Mi-F_1
Shimaoka et al. (2017)	51.7	70.9	64.9
AFET (Ren et al., 2016a)	55.1	71.1	64.7
PLE (Ren et al., 2016b)	57.2	71.5	66.1
BERT	51.8	76.6	69.1
MULTITASK	59.5	76.8	71.8
LABELGCN	**59.6**	**77.8**	**72.2**
HY LARGE	47.4	75.8	69.4

Table 7: Total accuracy, macro and micro F_1 scores on OntoNotes test set.

and Barbosa (2018) model the relations through a hierarchy-aware loss function. Xiong et al. (2019) derive a graph from type co-occurrence statistics in the dataset. Experimental evidence suggests that our model encodes similar hierarchical information without the need to provide it explicitly.

Hyperbolic representations have been employed for Question Answering (Tay et al., 2018), in Machine Translation (Gulcehre et al., 2019), and modeling language (Dhingra et al., 2018; Tifrea et al., 2019). We build upon the hyperbolic neural layers introduced in Ganea et al. (2018), and develop the missing components to perform, not binary, but multi-class multi-label text classification. We test the proposed model not with a synthetic dataset, but on a concrete downstream tasks, such as entity typing. Our work resembles López et al. (2019) and Chen et al. (2019), though they separately learn embeddings for type labels and text representations in hyperbolic space, whereas we do it in an integrated fashion.

8 Conclusions

Incorporating hierarchical information from the label inventory into neural models has become critical to improve performance. Hyperbolic spaces are an exciting approach since they are naturally equipped to model hierarchical structures. However, previous work integrated isolated components into neural systems. In this work we propose a fully hyperbolic model and showcase its effectiveness on challenging datasets. Our hyperbolic model automatically infers the latent hierarchy from the class distribution, captures implicit hyponymic relations in the inventory and achieves a performance comparable to state-of-the-art systems on very fine-grained labels with a remarkable reduction of the parameter size. This emphasizes the importance of choosing a metric space suitable to the data distribution as an effective inductive bias to capture fundamental properties, such as hierarchical structure.

Moreover, we illustrate ways to integrate different components with Euclidean layers, showing their strengths and drawbacks. An interesting future direction is to employ hyperbolic representations in combination with contextualized word embeddings. We release our implementation with the aim to ease the adoption of hyperbolic components into neural models, yielding lightweight and efficient systems.

Acknowledgments

This work has been supported by the German Research Foundation (DFG) as part of the Research Training Group Adaptive Preparation of Information from Heterogeneous Sources (AIPHES) under grant No. GRK 1994/1 and the Klaus Tschira Foundation, Heidelberg, Germany.

References

Abhishek Abhishek, Ashish Anand, and Amit Awekar. 2017. Fine-grained entity type classification by jointly learning representations and label embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 797–807, Valencia, Spain. Association for Computational Linguistics.

Gary Bécigneul and Octavian-Eugen Ganea. 2019. Riemannian adaptive optimization methods. In *7th International Conference on Learning Representations, ICLR*, New Orleans, LA, USA.

James W. Cannon, William J. Floyd, Richard Kenyon, and Walter R. Parry. 1997. *Hyperbolic Geometry*, volume 31. Flavors of Geometry.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems 32*, pages 4869–4880. Curran Associates, Inc.

Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, and Liping Jing. 2019. Hyperbolic interaction model for hierarchical multi-label classification. *CoRR*, abs/1905.10802.

Hyunghoon Cho, Benjamin DeMeo, Jian Peng, and Bonnie Berger. 2019. Large-margin classification in hyperbolic space. In *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1832–1840. PMLR.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia. Association for Computational Linguistics.

Luciano Del Corro, Abdalghani Abujabal, Rainer Gemulla, and Gerhard Weikum. 2015. FINET: Context-aware fine-grained named entity typing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 868–878, Lisbon, Portugal. 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 *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Octavian Ganea, Gary Becigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems 31*, pages 5345–5355. Curran Associates, Inc.

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-Dependent Fine-Grained Entity Type Tagging. *ArXiv e-prints*.

Caglar Gulcehre, Misha Denil, Mateusz Malinowski, Ali Razavi, Razvan Pascanu, Karl Moritz Hermann, Peter Battaglia, Victor Bapst, David Raposo, Adam Santoro, and Nando de Freitas. 2019. Hyperbolic attention networks. In *7th International Conference on Learning Representations, ICLR*, New Orleans, LA, USA.

Max Kochurov, Rasul Karimov, and Sergei Kozlukov. 2020. Geoopt: Riemannian optimization in PyTorch. *ArXiv*, abs/2005.02819.

Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguñá. 2010. Hyperbolic geometry of complex networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 82:036106.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 94–100. AAAI Press.

Qi Liu, Maximilian Nickel, and Douwe Kiela. 2019. Hyperbolic graph neural networks. In *Advances in Neural Information Processing Systems 32*, pages 8228–8239. Curran Associates, Inc.

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

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 97–109, Melbourne, Australia. Association for Computational Linguistics.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6341–6350. Curran Associates, Inc.

Maximillian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3779–3788, Stockholmsmässan, Stockholm Sweden. PMLR.

Yasumasa Onoe and Greg Durrett. 2019. Learning to denoise distantly-labeled data for entity typing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2407–2417, Minneapolis, Minnesota. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke

Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016a. AFET: Automatic fine-grained entity typing by hierarchical partial-label embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1369–1378, Austin, Texas. Association for Computational Linguistics.

Xiang Ren, Wenqi He, Meng Qu, Clare R. Voss, Heng Ji, and Jiawei Han. 2016b. Label noise reduction in entity typing by heterogeneous partial-label embedding. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1825–1834, New York, NY, USA. ACM.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, pages 69–74, San Diego, CA. Association for Computational Linguistics.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural architectures for fine-grained entity type classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1271–1280, Valencia, Spain. Association for Computational Linguistics.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 583–591, New York, NY, USA. ACM.

Alexandru Tifrea, Gary Becigneul, and Octavian-Eugen Ganea. 2019. Poincare Glove: Hyperbolic word embeddings. In *7th International Conference on Learning Representations, ICLR*, New Orleans, LA, USA.

Abraham Albert Ungar. 2008a. *Analytic Hyperbolic Geometry and Albert Einstein's Special Theory of Relativity*. World Scientific.

Abraham Albert Ungar. 2008b. *A Gyrovector Space Approach to Hyperbolic Geometry*. Morgan & Claypool.

Abraham Albert Ungar. 2010. *Barycentric Calculus in Euclidean and Hyperbolic Geometry: A Comparative Introduction*. World Scientific.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Wenhan Xiong, Jiawei Wu, Deren Lei, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Imposing label-relational inductive bias for extremely fine-grained entity typing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 773–784, Minneapolis, Minnesota. Association for Computational Linguistics.

Peng Xu and Denilson Barbosa. 2018. Neural fine-grained entity type classification with hierarchy-aware loss. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 16–25, New Orleans, Louisiana. Association for Computational Linguistics.

Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, and Gerhard Weikum. 2012. HYENA: Hierarchical type classification for entity names. In *Proceedings of COLING 2012: Posters*, pages 1361–1370, Mumbai, India.

A Basics of Riemannian Geometry

Manifold: a n-dimensional manifold $\mathcal{M}$ is a space that can locally be approximated by $\mathbb{R}^n$. It generalizes the notion of a 2D surface to higher dimensions. More concretely, for each point x on $\mathcal{M}$, we can find a *homeomorphism* (continuous bijection with continuous inverse) between a neighbourhood of x and $\mathbb{R}^n$.

Tangent space: the *tangent space* $T_x\mathcal{M}$ at a point x on $\mathcal{M}$ is a n-dimensional hyperplane in $\mathbb{R}^{n+1}$ that best approximates $\mathcal{M}$ around x. It is the first order linear approximation.

Riemannian metric: A *Riemannian metric* $g = (g_x)_{x\in\mathcal{M}}$ on $\mathcal{M}$ is a collection of inner-products $g_x : T_x\mathcal{M} \times T_x\mathcal{M} \to \mathbb{R}$ varying smoothly with x on tangent spaces. Riemannian metrics can be used to measure distances on manifolds

Riemannian manifold: is a pair $(\mathcal{M}, g)$, where $\mathcal{M}$ is a smooth manifold and $g = (g_x)_{x\in\mathcal{M}}$ is a Riemannian metric.

Geodesics: $\gamma : [0, 1] \to \mathcal{M}$ are the generalizations of straight lines to Riemannian manifolds, i.e., constant speed curves that are locally distance minimizing. In the Poincaré disk model, geodesics are circles that are orthogonal to the boundary of the disc as well as diameters.

Parallel transport: defined as $P_{x\to y} : T_x\mathcal{M} \to T_y\mathcal{M}$, is a linear isometry between tangent spaces that corresponds to moving tangent vectors along geodesics. It is a generalization of translation to non-Euclidean geometry, and it defines a canonical way to connect tangent spaces.

B Möbius Operations

Möbius scalar multiplication: for $x \in \mathbb{D}^n\backslash\{0\}$ the Möbius scalar multiplication by $r \in \mathbb{R}$ is defined as:

$$r \otimes x = \tanh(r \tanh^{-1}(\|x\|))\frac{x}{\|x\|} \tag{11}$$

and $r \otimes 0 := 0$. By making use of the exp and log maps, this expression is reduced to:

$$r \otimes x = \exp_{\mathbf{0}}(r \log_{\mathbf{0}}(x)), \quad \forall r \in \mathbb{R}, x \in \mathbb{D}^n \tag{12}$$

Exponential and logarithmic maps: The mapping between the tangent space and hyperbolic space is done by the exponential map $\exp_x : T_x\mathbb{D}^n \to \mathbb{D}^n$ and the logarithmic map $\log_x : \mathbb{D}^n \to T_x\mathbb{D}^n$. They are given for $v \in T_x\mathbb{D}^n\backslash\{0\}$ and $y \in \mathbb{D}^n\backslash\{0\}, y \neq x$:

$$\begin{aligned}\exp_x(v) &= x \oplus \left(\tanh\left(\frac{\lambda_x\|v\|}{2}\right)\frac{v}{\|v\|}\right)\\ \log_x(y) &= \frac{2}{\lambda_x}\tanh^{-1}(\|-x\oplus y\|)\frac{-x\oplus y}{\|-x\oplus y\|}\end{aligned} \tag{13}$$

These expressions become more appealing when $x = 0$, that is, at the origin of the space. It can be seen that the matrix-vector multiplication formula is derived from $M \otimes y = \exp_{\mathbf{0}}(M \log_{\mathbf{0}}(y))$. The point $y \in \mathbb{D}^n$ is mapped to the tangent space $T_{\mathbf{0}}\mathbb{D}^n$, the linear mapping M is applied in the Euclidean subspace, and finally the result is mapped back into the ball. A similar approach holds for the Möbius scalar multiplication and the application of pointwise non-linearity functions to elements in the Poincaré ball (see Ganea et al. (2018), Section 2.4).

Parallel transport with exp **and** log **maps:** By applying the exp and log maps the parallel transport in the Poincaré ball for a vector $v \in T_{\mathbf{0}}\mathbb{D}^n$ to another tangent space $T_x\mathbb{D}^n$, is given by:

$$P_{\mathbf{0}\to x}(v) = \log_x(x \oplus \exp_{\mathbf{0}}(v)) = \frac{\lambda_{\mathbf{0}}}{\lambda_x}v \tag{14}$$

This result is used to define and optimize the $a_k = (\lambda_0/\lambda_{p_k})a'_k$ in the Hyperbolic MLR (Appendix E)

C Hyperbolic Gated Recurrent Unit

To encode the context we apply a hyperbolic version of gated recurrent units (GRU) (Cho et al., 2014) proposed in Ganea et al. (2018):

$$\begin{aligned}r_t &= \sigma\left(log_0(W^r \otimes h_{t-1} \oplus U^r \otimes x_t \oplus b^r)\right)\\ z_t &= \sigma\left(log_0(W^z \otimes h_{t-1} \oplus U^z \otimes x_t \oplus b^z)\right)\\ \tilde{h_t} &= \tanh^{\otimes}((W diag(r_t)) \otimes h_{t-1} \oplus U \otimes x_t \oplus b)\\ h_t &= h_{t-1} \oplus diag(z_t) \otimes (-h_{t-1} \oplus \tilde{h_t})\end{aligned} \tag{15}$$

where $W \in \mathbb{R}^{d_S\times d_S}, U \in \mathbb{R}^{d_S\times n}, x_t \in \mathbb{D}^n$ and $b \in \mathbb{D}^{d_S}$ (superscripts are omitted). r_t is the reset gate, z_t is the update gate, $diag(x)$ denotes a diagonal matrix with each element of the vector x on its diagonal, and σ is the sigmoid function.

D Distance-based Attention

D.1 Formulation

In Equation 9 we calculate the Lorentz factors for each point x_i. The Lorentz factors are given by:

$$\gamma(x) = \frac{1}{\sqrt{1 - \|x\|^2}} \tag{16}$$

In the case of Gulcehre et al. (2019), the application of the Einstein midpoint (Ungar, 2010, Theorem 4.4) requires the mapping of the points onto the Klein model. By applying the Möbius midpoint, we avoid this mapping, and achieve an attention mechanism that operates only in one model of hyperbolic space.

D.2 Experimental Observations

To obtain the weights for the attention mechanism, initially Equation 8 was given by:

$$\alpha(q_i, k_i) = f(-\beta d_{\mathbb{D}}(q_i, k_i) - c) \tag{17}$$

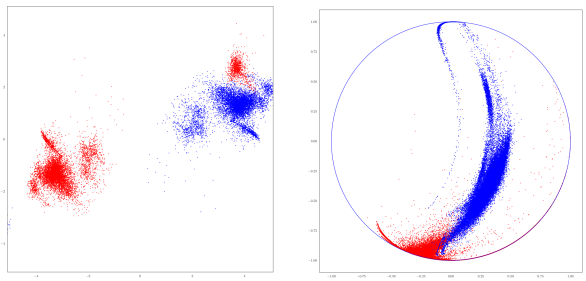
We experimented with replacing f for $\mathrm{sigmoid}$ and $\mathrm{softmax}$ functions. We found better performance with the latter one. Moreover, empirical observation lead us to remove the c value, since it converged to zero in all experiments. We believe that the biases b^Q and b^K from Equation 8 compensate for this c.

D.3 Queries and Keys

To further analyze the attention mechanism we plot the query q_i and key k_i points of Equation 8 for both models in Figure 5. It must be recalled that the shorter the distance between points, the higher the attention weight that the word gets assigned. Furthermore, we observed that the attention gets prominently centered on the mention in both models, assigning very low weights on the rest of the words in the context.

In the Euclidean space we can clearly distinguish the two clusters which make the distance-based attention to give very low weights on most words of the context. The small red cluster on the top right of the image belongs to points corresponding to words in the mention span. These words get projected very close to the key vector, in order to minimize the distance and increase the attention weight.

On the hyperbolic model, the queries get clustered at the bottom of the plot, whereas the keys are the points adjusting the distance to define the weight on each word.



(a) Euclidean Space. (b) Hyperbolic Space.

Figure 5: Queries (red) and keys (blue) projected in 2D for different spaces.

E Multinomial Logistic Regression

E.1 Hyperbolic MLR

The original formula from Ganea et al. (2018) for MLR in the Poincaré ball, given K classes and $k \in \{1, ..., K\}$, $p_k \in \mathbb{D}^n$, $a_k \in T_{p_k}\mathbb{D}^n \backslash \{0\}$, the formula for the hyperbolic MLR is:

$$p(y = k|x) \propto f\left(\frac{\lambda_{p_k}^c \|a_k\|}{\sqrt{c}} \sinh^{-1}\left(\frac{2\sqrt{c}\langle -p_k \oplus x, a_k\rangle}{(1 - c\| - p_k \oplus x\|^2)\|a_k\|}\right)\right) \tag{18}$$

Where $x \in \mathbb{D}^n$, p_k and a_k are trainable parameters, and c is a parameter in relation to the radius of the Poincaré ball $r = 1/\sqrt{c}$ which in this work we assume to be $c = 1$, hence it is omitted of the formulations. Since $a_k \in T_{p_k}\mathbb{D}^n$ and therefore depends on p_k, it is unclear how to perform optimization. The solution proposed by Ganea et al. (2018) is to re-express it as:

$$a_k = P_{\mathbf{0} \to p_k}(a'_k) = \frac{\lambda_{\mathbf{0}}}{\lambda_{p_k}} a'_k \tag{19}$$

where $a'_k \in T_{\mathbf{0}}\mathbb{D}^n = \mathbb{R}^n$. In this way we can optimize a'_k as a Euclidean parameter. Finally, when we use a'_k instead of a_k, the formula for the MLR is:

$$p(y = k|x) \propto f\left(2\|a'_k\| \sinh^{-1}\left(\frac{2\langle -p_k \oplus x, a'_k\rangle}{(1 - \| - p_k \oplus x\|^2)\|a'_k\|}\right)\right) \tag{20}$$

E.2 Euclidean MLR

The Euclidean formulation of the MLR is given by:

$$p(y = k|x) \propto f(4\langle -p_k \oplus x, a_k\rangle) \tag{21}$$

This equation arise from taking the limit of $c \to 0$ in Equation 18. In that case, $f(4\langle -p_k \oplus x, a_k\rangle) = f((\lambda_{p_k}^0)^2\langle -p_k \oplus x, a_k\rangle) = f(\langle -p_k \oplus x, a_k\rangle_0)$.

F Experimental Details

For the context-GRU we use $\tanh$ as non-linearity to establish a fair comparison against the classical GRU (Cho et al., 2014). On the char-RNN we use the $\mathrm{identity}$ (no non-linearity). The MLR is fed with the final representation achieved by

the concatenation of mention and context features: $\text{concat}(\mathbf{M}, \mathbf{C}, \mathbf{S}) \in \mathbb{D}^m$ with $m = d_M + d_C + 2d_S$.

In the XLARGE model, we use the Euclidean encoder in all experiments given time constraints.

Hyperparameters: Both hyperbolic and Euclidean models were trained with the hyperparameters detailed in Table 8.

Dropout: We apply low values of dropout given that the model was very sensitive to the this parameter. We consider this a logical behaviour since the distances in hyperbolic space grow exponentially with the norm of the points, making the model very responsive to this parameter.

Numerical Errors: they appear when the norm of the hyperbolic vectors is very close to 1 or 0. To avoid them we follow the recommendations reported on Ganea et al. (2018). The result of hyperbolic operations is always projected in the ball of radius $1 - \epsilon$, where $\epsilon = 10^{-5}$. When vectors are very close to 0, they are perturbed with an $\varepsilon = 10^{-15}$ before they are used in any of the above operations. Finally, arguments of the $tanh$ function are clipped between ± 15, while arguments of $tanh^{-1}$ are clipped in the interval $[-1 + 10^{-15}, 1 - 10^{-15}]$. Finally, and by recommendations of the Geoopt developers (Kochurov et al., 2020), we operate on floating point of 64 bits.

Initialization: we initialize character and positional embeddings randomly from the uniform distribution $U(-0.0001, 0.0001)$. In the case of the hyperbolic model, we map them into the ball with the $\exp_{\mathbf{0}}$ map. We initialize all layers in the model using *Glorot uniform initialization*.

Exponential and logarithmic map: In the case of the Glove embedding ablation (Section 6.1.1), we used the $100d$ version, trained over Wikipedia and Gigaword[4]. By directly applying the logarithmic map, the embeddings were projected close to the border of the ball, making the model very unstable. To overcome this, we use a parameter c described in Ganea et al. (2018) to adjust the radius of the ball, which helps to project the embeddings closer to the origin of the space.

Hardware: All experiments for the hyperbolic and Euclidean models were performed using 2 NVIDIA P40 GPUs, with the batch sizes specified in Table 8.

[4] http://nlp.stanford.edu/data/glove.6B.zip

Parameter	Value
Batch size BASE	900
Batch size LARGE	350
Batch size XLARGE	160
BASE d_M	40
BASE d_C	20
BASE d_S	20
BASE $d_M + d_C + 2d_S$	100
LARGE d_M	100
LARGE d_C	50
LARGE d_S	50
LARGE $d_M + d_C + 2d_S$	250
XLARGE d_M	200
XLARGE d_C	100
XLARGE d_S	100
XLARGE $d_M + d_C + 2d_S$	500
Mention non-linearity	tanh
Context non-linearity	tanh
Epochs	40
Crowd cycles	5
Input dropout	0.2
Concat dropout	0.1
Learning rate	0.0005
Weight decay	0.0
Max. gradient norm	5

Table 8: Hyperparameters of the models.

G Closest Types

We report the points p_k learned by the model to define the hyperplanes of Equation 10. Table 9 shows the types corresponding to the closest points, measured by their hyperbolic distance $d_{\mathbb{D}}$ (see Eq 1), to the *coarse* labels. We observe that the types are highly correlated given that they often co-occur in the same context.

H More Experimental Observations

Text vectors norms: By "text vector" we refer the concatenated vector of the context, mention and char-level mention representations before the MLR layer. We report the average norm of this vectors per training epoch, for the 20D Euclidean and hyperbolic model on Figure 6. The norm of the vectors of the hyperbolic model are measured according to the hyperbolic distance $d_{\mathbb{D}}$ (see Equation 1). That is, we take the hyperbolic distance from the origin to the point, thus the values are above one. The norm of the Euclidean model is measured according to the Euclidean norm. We

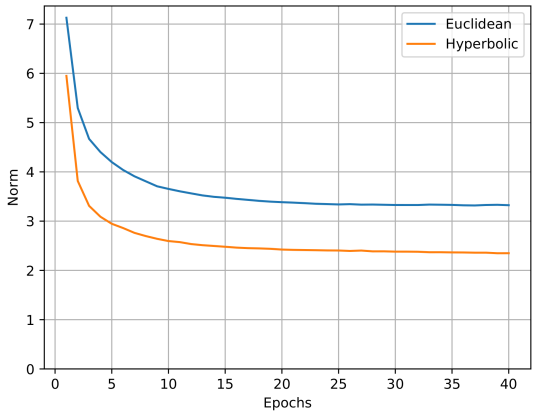


Figure 6: Norm of *text vectors* for the Euclidean and hyperbolic model. The hyperbolic norm is measured as the hyperbolic distance $d_{\mathbb{D}}$ from the origin to the point, hence the values can be greater than 1.

observe that both models learn to reduce the norm of the vectors, and it is noticeable that the convergence value for the Euclidean model is higher than for the hyperbolic model.

organization		institution		firm		group		unit		division	
Types	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$
institution	0.34	firm	0.24	business	0.23	unit	0.34	division	0.26	subsidiary	0.25
company	0.35	company	0.26	institution	0.24	gathering	0.34	theatre	0.28	unit	0.26
news_agency	0.36	university	0.26	company	0.25	subject	0.34	activist	0.28	track	0.28
business	0.38	operator	0.28	maker	0.27	administration	0.36	position	0.28	half	0.28
administration	0.40	maker	0.28	operator	0.28	affiliation	0.36	half	0.28	activist	0.29
location		state		country		place		space		half	
Types	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$
state	0.33	country	0.29	state	0.31	space	0.40	half	0.28	peak	0.26
cemetery	0.35	half	0.31	nation	0.31	localization	0.40	shopping_mall	0.29	operator	0.26
space	0.35	agency	0.31	agency	0.32	place_name	0.40	venue	0.29	theatre	0.26
half	0.35	activist	0.32	kingdom	0.34	close	0.41	landmark	0.30	placement	0.26
area	0.36	unit	0.32	world	0.35	birthplace	0.41	localization	0.30	summit	0.26
event		conflict		war		time		duration		calendar	
Types	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$
conflict	0.44	war	0.34	guerrilla	0.32	duration	0.40	calendar	0.30	date	0.22
activist	0.45	dispute	0.36	conflict	0.34	period	0.43	peak	0.31	phrase	0.25
election	0.45	series	0.37	military	0.35	length	0.46	half	0.32	second	0.26
activity	0.46	guerrilla	0.38	citizen	0.36	month	0.46	second	0.32	activist	0.27
holiday	0.46	future	0.38	situation	0.36	date	0.46	fantasy	0.32	need	0.28
object		machine		computer		entity		separation		placement	
Types	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$	**Types**	$d_{\mathbb{D}}$
machine	0.37	computer	0.29	version	0.29	separation	0.43	placement	0.27	position	0.25
arrangement	0.39	theatre	0.30	machine	0.30	relative	0.44	missionary	0.27	localization	0.26
medium	0.39	operator	0.30	communication	0.30	meaning	0.44	meaning	0.27	half	0.26
method	0.39	card_game	0.31	activist	0.31	warlord	0.45	variation	0.27	separation	0.27
representation	0.39	core	0.31	maker	0.32	baseball	0.45	phrase	0.27	winner	0.27

Table 9: Closest p_k points in the Poincaré Ball to *coarse* entity types, with their hyperbolic distance. In many cases, a hierarchical relation holds with the closest type. For example: `firm` *is-a* `institution` *is-a* `organization`.