Skip-gram word embeddings in hyperbolic space

Matthias Leimeister[*]
Lateral GmbH
Berlin, Germany
matthias@lateral.io

Benjamin J. Wilson[*]
Lateral GmbH
Berlin, Germany
benjamin@lateral.io

Abstract

Embeddings of tree-like graphs in hyperbolic space were recently shown to surpass their Euclidean counterparts in performance by a large margin. Inspired by these results, we present an algorithm for learning word embeddings in hyperbolic space from free text. An objective function based on the hyperbolic distance is derived and included in the skip-gram architecture from word2vec. The hyperbolic word embeddings are then evaluated on word similarity and analogy benchmarks. The results demonstrate the potential of hyperbolic word embeddings, particularly in low dimensions, though without clear superiority over their Euclidean counterparts. We further discuss problems in the formulation of the analogy task resulting from the curvature of hyperbolic space.

1 Introduction

Machine learning algorithms are often based on features in Euclidean space, assuming a flat geometry. For example, word embeddings trained by the word2vec algorithm represent each word as a finite-dimensional vector, with the inner product serving as a similarity measure during training. However, in many applications there is a more natural representation of the underlying data in terms of a curved manifold. Hyperbolic space is a negatively-curved, non-Euclidean space. It is advantageous for embedding trees as the circumference of a circle grows exponentially with the radius. Learning embeddings in hyperbolic space has recently gained interest (Nickel and Kiela, 2017; Chamberlain et al., 2017; De Sa et al., 2018). So far most works on hyperbolic embeddings have dealt with network or tree-like data and focused on link reconstruction or prediction as evaluation measures. This led us to consider hyperbolic space for training word embeddings using the skip-gram model. As with the graph embeddings, the exponential growth of the circumference in hyperbolic space should give different words space to spread out, even in low dimensions.

The contributions of this report are the proposition of an objective function for skip-gram on the hyperboloid model of hyperbolic space, the derivation of update equations for gradient based optimization, first experiments on common word embedding evaluation tasks and a discussion of the difficulty of adapting the analogy task to manifolds with curvature.

The paper is structured as follows. In section 2, we summarize prior work on word vector representations and recent works on hyperbolic graph embeddings. In section 3, we introduce notations from Riemannian geometry and describe the hyperboloid model of hyperbolic space. Section 4 reviews the skip-gram architecture from word2vec and suggests an objective function for learning word embeddings on the hyperboloid. In section 5, we evaluate the proposed architecture for common word similarity and analogy tasks and compare the results with the standard Euclidean skip-gram algorithm.

2 Related work

Learning semantic representations of words has long been a focus of natural language processing research. Early models for dense vector representations included Latent Semantic Indexing (LSI) (Deerwester et al., 1990), where a word-context matrix is decomposed by singular value decomposition to produce low dimensional embedding vectors. A probabilistic framework based on topic modeling that also produces dense word vectors was introduced by Latent Dirichlet Analysis (LDA) (Blei et al., 2003). Neural network

[*]Authors contributed equally.

models for word embeddings have first emerged in the context of language modeling (Bengio et al., 2003; Mnih and Hinton, 2008), where word embeddings are learned as intermediate features of a neural network predicting the next word from a sequence of past words. The word2vec algorithm, introduced in Mikolov et al. (2013), aimed instead to learn word embeddings that would be useful for a broader range of downstream tasks. The seminal word2vec paper also introduced a new intrinsic evaluation method for word embeddings, the so-called analogy task. It measures the extent to which word relationships are captured arithmetically by the embedding space (see section 5 for more details).

The use of hyperbolic geometry for learning embeddings has recently received some attention in the field of graph embeddings. Nickel and Kiela (2017) use the Poincaré ball model of hyperbolic space and an objective function based on the hyperbolic distance to embed the vertices of a tree derived from the WordNet "is-a" relations. They report far superior performance in terms of graph reconstruction and link prediction compared to the same embedding method in a Euclidean space of the same dimension. Chamberlain et al. (2017) use the Euclidean scalar product rescaled by the hyperbolic distance from the origin as a similarity function for an embedding algorithm and report qualitatively better embeddings of different graph datasets compared to Euclidean space. This amounts to pulling back all data points to the tangent space at the origin and then optimizing in this tangent space. Finally, De Sa et al. (2018) present a combinatorial algorithm for embedding graphs in the Poincaré ball that outperforms prior algorithms and can be tuned using the allowed precision versus the expected distortion of the embeddings. In a follow-up paper to the Poincaré embeddings, Nickel and Kiela (2018) use the hyperboloid model in Minkowski space to learn graph embeddings and show its benefits for gradient based optimization. As we work in the same model of hyperbolic space, their derivation of the update equation is largely similar to ours. Finally, one other recent paper deals with learning hyperbolic embeddings for words and sentences from free text. Dhingra et al. (2018) construct a layer on top of a neural network architecture that maps the preceding activations to polar coordinates on the Poincaré disk. For learning word embeddings, a co-occurrence graph is constructed and embeddings are learned using the algorithm from Nickel and Kiela (2017). Their evaluation shows that the resulting hyperbolic embeddings perform better on inferring lexical entailment relations than Euclidean embeddings trained with skip-gram. However, their hyperbolic embeddings show no advantage for standard word similarity tasks. Moreover, in order to compare the similarity of two words, the authors use the cosine similarity, which is inconsistent with the hyperbolic geometry.

3 Geometry of hyperbolic space

The following sections introduce the hyperboloid model of hyperbolic space together with the explicit formulation of some core concepts from Riemannian geometry. For a general introduction to Riemannian manifolds see e.g. Petersen (2006).

We identify points in Euclidean or Minkowski space with their position vectors and denote both by lower case letters. Coordinate components are denoted by lower indexes, as in v_i. For a non-zero vector v in a normed vector space, $\hat{v}$ denotes its normalization, i.e. $\hat{v} = \frac{v}{\|v\|}$.

3.1 The hyperboloid model in Minkowski space

The relationship of the hyperboloid to its ambient space, called Minkowski space, is analogous to that between the sphere and its ambient Euclidean space. For a detailed account of the hyperboloid model, see e.g. Reynolds (1993).

Definition 3.1. The $(n+1)$-dimensional *Minkowski space* $\mathbb{R}^{(n,1)}$ is the real vector space $\mathbb{R}^{n+1}$ endowed with the *Minkowski dot product*:

$$\langle u, v \rangle_{Mink} = \sum_{i=0}^{n-1} u_i v_i - u_n v_n, \tag{1}$$

for $u, v \in \mathbb{R}^{(n,1)}$.

The Minkowski dot product is not positive definite, i.e. there are vectors for which $\langle v, v \rangle_{Mink} < 0$. Therefore, Minkowski space is not an inner product space. A common usage of the Minkowski space $\mathbb{R}^{(3,1)}$ is in special relativity, where the first three (Euclidean) dimensions represent space, and the last time. One common model of hyperbolic space is as a subset of Minkowski space in the form of the upper sheet of a two-sheeted hyperboloid.

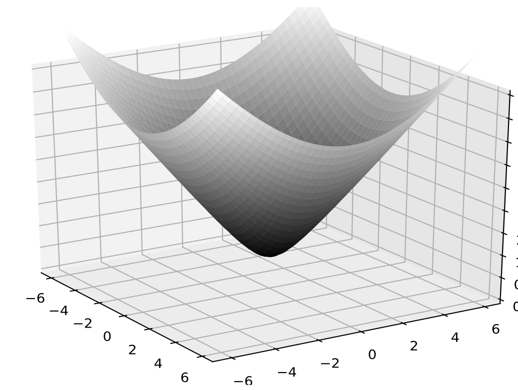

Figure 1: Hyperbolic space as the upper sheet of a hyperboloid in Minkowski space.

Definition 3.2. The *hyperboloid model* of hyperbolic space is defined by

$$\mathbb{H}^n = \{\, x \in \mathbb{R}^{(n,1)} \mid \langle x, x \rangle_{Mink} = -1,\, x_n > 0 \,\}. \tag{2}$$

The tangent space at a point $p \in \mathbb{H}^n$ is denoted by $T_p\mathbb{H}^n$. It is the orthogonal complement of p with respect to the Minkowski dot product:

$$T_p\mathbb{H}^n = \{\, x \in \mathbb{R}^{(n,1)} \mid \langle x, p \rangle_{Mink} = 0 \,\}.$$

$\mathbb{H}^n$ is a smooth manifold and can be equipped with a Riemannian metric by the induced scalar product from the ambient Minkowski dot product on the tangent spaces:

$$\begin{aligned} \text{For } p \in \mathbb{H}^n, \quad & v, w \in T_p\mathbb{H}^n, \\ g_p(v, w) &:= \langle v, w \rangle_{Mink}. \end{aligned} \tag{3}$$

The restriction of the Minkowski dot product yields a positive definite inner product on the tangent spaces of $\mathbb{H}^n$, despite not being positive definite itself. This makes $\mathbb{H}^n$ a Riemannian manifold.

3.2 Optimization in hyperbolic space

Similar to a model in Euclidean space, stochastic gradient descent can be used to find local minima of a differentiable objective function $f : \mathbb{H}^n \to \mathbb{R}$. However, since hyperbolic space is a Riemannian manifold, the gradient of the function at a point $p \in \mathbb{H}^n$ will be an element of the tangent space $T_p\mathbb{H}^n$. Therefore, adding the gradient to the current parameter does not produce a point in $\mathbb{H}^n$, but in the ambient space $\mathbb{R}^{(n,1)}$. There are several approaches to still use additive updates as an approximation. However, Bonnabel (2011) presents Riemannian gradient descent as a way to use the geometric structure in order to make mathematically sound updates. Furthermore, Wilson and Leimeister (2018) illustrate the benefit of using Riemannian gradient descent in hyperbolic space instead of first-order approximations using retractions. The updates use the so-called exponential map, Exp_p, which maps a tangent vector $v \in T_p\mathbb{H}^n$ to a point on $\mathbb{H}^n$ that is $\|v\|$ far away from p in the direction of v. First, the gradient of the loss function with respect to a parameter is computed. Then the parameter is updated by applying the exponential map to the negative gradient vector scaled by a learning rate:

$$x^{new} = \mathrm{Exp}_{x^{old}}(-\eta \nabla_{x^{old}} f(x^{old})). \tag{4}$$

The shortest paths that are mapped out by the exponential map are called geodesic curves. The geodesics of $\mathbb{H}^n$ are its intersections with two-dimensional planes through the origin. For a point $p \in \mathbb{H}^n$ and an initial direction $v \in T_p\mathbb{H}^n$ with $\langle v, v \rangle_{Mink} = 1$, the geodesic curve is given by

$$\gamma_{p,v} : \mathbb{R} \to \mathbb{H}^n,\ \gamma_{p,v}(t) = \cosh(t) \cdot p + \sinh(t) \cdot v. \tag{5}$$

The *hyperbolic distance* for two points $p, q \in \mathbb{H}^n$ is computed by

$$d_{\mathbb{H}^n}(p, q) = \operatorname{arccosh}(-\langle p, q \rangle_{Mink}). \tag{6}$$

The closed form formulas for geodesics and the hyperbolic distance make the hyperboloid model attractive for formulating optimization problems. This is a clear advantage compared to other models of hyperbolic space, where equations take a more complicated form (c.f. the hyperbolic distance and update equations on the Poincaré ball in Nickel and Kiela (2017)).

3.3 Parallel transport along geodesics in $\mathbb{H}^n$

In order to carry out the analogy task on $\mathbb{H}^n$, parallelograms need to be constructed. This is achieved by parallel transport along geodesics. Parallel transport provides a way to identify the tangent spaces and move a vector from one tangent space to another along a geodesic curve while preserving angles and length.

To derive a formula for the parallel transport of a vector $w \in T_p\mathbb{H}^n$ along a geodesic γ starting at $p \in \mathbb{H}^n$ in direction $v \in T_p\mathbb{H}^n$, we decompose w into its component along $\hat{v}$ and the orthogonal complement:

$$w = \langle w, \hat{v} \rangle \hat{v} + (w - \langle w, \hat{v} \rangle \hat{v}),$$

where we denote by $\langle \cdot, \cdot \rangle$ the Minkowski dot product for brevity. The geodesic is given by

$$\gamma(t) = \mathrm{Exp}_p(t \cdot v) = \cosh(t\|v\|) \cdot p + \sinh(t\|v\|)\hat{v}$$

The first summand of the initial decomposition is parallel to the tangent vector $\gamma'(0)$ of the geodesic. Parallel transport will keep this part parallel to the tangent vector of γ for all t. Therefore, it is mapped to $\langle w, \hat{v} \rangle \gamma'(1)$.

The derivative at the end point is

$$\gamma'(1) = \sinh(\|v\|) \cdot p + \cosh(\|v\|) \cdot \hat{v},$$

which is an element of the linear subspace spanned by $\{p, v\}$. For the orthogonal component we fix a basis that completes $\{p, v\}$ to a basis of $\mathbb{R}^{(n,1)}$. This part of the decomposition is constant when parallel transported along the curve. This gives the following closed form solution for parallel transport along a geodesic in $\mathbb{H}^n$.

Theorem 3.1. Let $p \in \mathbb{H}^n$ be a point on the hyperboloid and $v, w \in T_p\mathbb{H}^n$. The parallel transport of w along the geodesic in direction v is given by

$$\varphi_{p,v}(w) = \langle w, \hat{v} \rangle \cdot (\sinh(\|v\|) \cdot p + \cosh(\|v\|) \cdot \hat{v}) + w - \langle w, \hat{v} \rangle \cdot \hat{v}. \tag{7}$$

One can verify that with the above formula for parallel transport, there holds $\langle \varphi_{p,v}(w_1), \varphi_{p,v}(w_2) \rangle = \langle w_1, w_2 \rangle$ for all $p \in \mathbb{H}^n$ and $v, w_1, w_2 \in T_p\mathbb{H}^n$.

4 Hyperbolic skip-gram model

4.1 word2vec skip-gram

The skip-gram architecture was first introduced by Mikolov et al. (2013) as one version of the *word2vec* framework. Given a continuous stream of text

$$T = (w_0, w_1, w_2, \ldots)$$

with words w_i from a fixed vocabulary $\mathcal{V}$. Skip-gram training learns a vector representation in Euclidean space for each word. This representation captures meaning in the sense that words with similar co-occurrence distributions map to nearby vectors. Given a center word w_0 and a context of surrounding words

$$\mathcal{C} = (w_{-n}, \ldots, w_{-1}, w_1, \ldots, w_n)$$

the task in skip-gram learning is to predict each context word from the center word. One way to efficiently train these embeddings is *negative sampling*, where the embeddings are optimized to identify which of a selection of vocabulary words likely occurred as context words (Mikolov et al., 2013b).

The center and context words are parametrized as two layers of a neural network architecture. The first layer, representing the center words, is given by the parameter matrix $\alpha \in \mathbb{R}^{d \times |\mathcal{V}|}$, with $|V|$ being the number of words in the vocabulary, and d the embedding dimension. Similarly, the output layer is given by $\gamma \in \mathbb{R}^{d \times |\mathcal{V}|}$. For both, the columns are indexed by words from the vocabulary $w \in \mathcal{V}$, i.e. $\alpha_w, \gamma_w \in \mathbb{R}^d$.

Let $u \in \mathcal{V}$ be the center word and $w_0 \in \mathcal{V}$ be a context word. Negative sampling training then chooses a number of noise samples $\{w_1, \ldots, w_k\}$. The objective function to maximize for this combination of center and context word is then

$$\mathcal{L}_{u,w_0}(\alpha, \gamma) = \prod_{i=0}^{k} P(y_i | w_i, u) = \prod_{i=0}^{k} \sigma((-1)^{1-y_i} \langle \alpha_u, \gamma_{w_i} \rangle_{\mathbb{R}^d}), \tag{8}$$

with the labels

$$y_i = \begin{cases} 1 & \text{if } i = 0 \\ 0 & \text{else.} \end{cases}$$

Each such step in skip-gram learning can be understood as training a logistic regression classifier that labels the true context word from negative samples. The parameters α and γ are optimized using stochastic gradient descent on the negative log likelihood. After training, the vectors of one parameter matrix (in common implementations the input layer, although other publications use both layers, or an aggregate thereof) are the resulting *word embeddings* and can be used as features in downstream tasks.

4.2 An objective function for skip-gram training on the hyperboloid

The Euclidean inner product in the skip-gram objective function represents the similarity measure for two word embeddings. Thus, co-occurring words should have a high dot product.

Similarly, in hyperbolic space, one can define a similarity by requiring that similar words have a low hyperbolic distance. Since $\operatorname{arccosh}$ is monotone, the hyperbolic distance from equation 6 is proportional to the negative Minkowski dot product. This yields an efficient way to represent the similarity on the hyperboloid by just using the Minkowski dot product as similarity function. However, the Minkowski dot product between two points on the hyperboloid is bounded above by -1 (reaching the upper bound if and only if the two points are equal). Therefore, when using it as a similarity function in the likelihood function, we apply an additive shift to center the values around 0:

$$P(y|w,u) = \sigma\left((-1)^{1-y}(\langle \alpha_u, \gamma_w \rangle_{Mink} + \theta)\right) \tag{9}$$

θ is either an additional hyperparameter or could be learned during training. The full loss function for a center word u, context word w_0, and negative samples $\{w_1, \dots, w_n\}$ is similar to equation 8:

$$\mathcal{L}_{u,w_0}(\alpha,\gamma) = \prod_{i=0}^{k} P(y_i|w_i,u) = \prod_{i=0}^{k} \sigma\left((-1)^{1-y_i}(\langle \alpha_u, \gamma_{w_i} \rangle_{Mink} + \theta)\right) \tag{10}$$

By using $\langle p,q \rangle_{Mink} = -\cosh(d_{\mathbb{H}^n}(p,q))$, the objective function for a positive sample ($y=1$) can be evaluated in terms of the hyperbolic distance between two points in $\mathbb{H}^n$. This leads to the function depicted in Figure 2. The choice of the hyperparameter θ affects the onset of the decay that leads to an upper bound on the hyperbolic distance after which a negative sample will not influence the updates anymore. This leads to optimizing for a margin between an input word and negative samples. After the margin is reached the negative samples are considered sufficiently far away.

4.3 Geodesic update equations

To compute the derivatives of the objective function, we will use the following

Lemma 4.1. For a differentiable function $f : \mathbb{R}^{(n,1)} \to \mathbb{R}$, the gradient is given by

$$\nabla f = \left(\frac{\partial f}{\partial x_0}, \dots, \frac{\partial f}{\partial x_{n-1}}, -\frac{\partial f}{\partial x_n}\right), \tag{11}$$

where the $\frac{\partial f}{\partial x_i}$ denote partial derivatives according to the Euclidean vector space structure of $\mathbb{R}^{(n,1)}$.

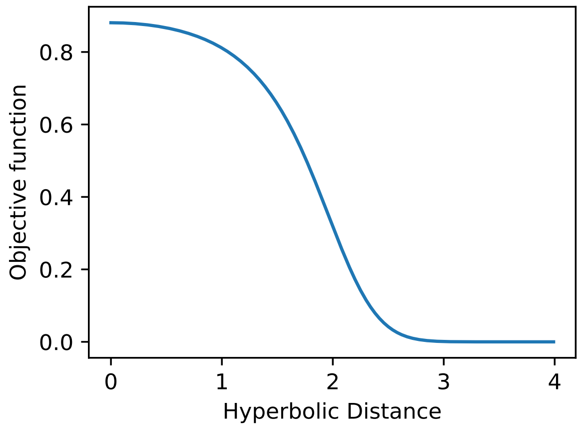

Figure 2: Values of the likelihood for a positive sample in terms of the hyperbolic distance, $\theta = 3$.

In order to compute the Riemannian gradient of the objective function $\mathcal{L}$, we first compute the gradient of $\mathcal{L}$ extended to $\mathbb{R}^{(n,1)}$ according to equation 11. Then the Riemannian gradient is the projection of this gradient to the tangent space $T_p\mathbb{H}^n$ at the parameter point $p \in \mathbb{H}^n$. For the first layer parameters we get

$$\nabla_{\alpha_u}^{\mathbb{R}^{(n,1)}} \log \mathcal{L}_{u,w_0}(\alpha,\gamma) = \sum_{i=0}^{k} (y_i - \sigma(\langle \alpha_u, \gamma_{w_i} \rangle_{Mink} + \theta)) \cdot \gamma_{w_i}. \tag{12}$$

In a similar fashion, one can compute the gradient for a second layer parameter γ_w. For this, let $\mathcal{S}_u := \{w_0, w_1, \dots, w_k\}$ be the set of positive and negative samples for the present update step and denote by $\#_{w,\mathcal{S}_u}$ the count of a word w in $\mathcal{S}$. Furthermore, let

$$y(w) = \begin{cases} 1 & \text{if } w = w_0 \\ 0 & \text{if } w \in \{w_1, \dots, w_k\}. \end{cases}$$

Then the gradient is given by

$$\nabla_{\gamma_w}^{\mathbb{R}^{(n,1)}} \log \mathcal{L}_{u,w_0}(\alpha,\gamma) = \#_{w,\mathcal{S}_u} (y(w) - \sigma(\langle \alpha_u, \gamma_{w_i} \rangle_{Mink} + \theta)) \cdot \alpha_u. \tag{13}$$

Finally both gradients are projected onto the tangent space of $\mathbb{H}^n$ via

Theorem 4.2. Let $p \in \mathbb{H}^n$ and $v \in \mathbb{R}^{(n,1)}$. Then the projection of v onto $T_p\mathbb{H}^n$ is given by

$$\mathrm{proj}_p(v) = v + \langle p, v \rangle_{Mink} \cdot p. \tag{14}$$

The resulting projections give the Riemannian gradients on $\mathbb{H}^n$,

$$\nabla^{\mathbb{H}^n}_{\gamma_w} \log \mathcal{L}_{u,w_0}(\alpha,\gamma) = \operatorname{proj}_{\gamma_w}\left(\nabla^{\mathbb{R}^{(n,1)}}_{\gamma_w} \log \mathcal{L}_{u,w_0}(\alpha,\gamma)\right) \tag{15}$$

$$\nabla^{\mathbb{H}^n}_{\alpha_u} \log \mathcal{L}_{u,w_0}(\alpha,\gamma) = \operatorname{proj}_{\alpha_u}\left(\nabla^{\mathbb{R}^{(n,1)}}_{\alpha_u} \log \mathcal{L}_{u,w_0}(\alpha,\gamma)\right) \tag{16}$$

that are used for Riemannian stochastic gradient descent according to equation 4.

5 Experiments

In order to evaluate the quality of the learned embeddings, various common benchmark datasets are available. On the word level, two popular tasks are word similarity and analogy. These will be used here to compare the hyperbolic embeddings with their Euclidean counterparts.

5.1 Training setup

Word embeddings are trained on a 2013 dump of Wikipedia that has been preprocessed by lower casing and removing punctuation, and filtered to only contain articles with at least 20 page views[1]. This results in a corpus of 463k documents with 498 Million words. For learning word embeddings in Euclidean space we use the skip-gram implementation of *fastText*[2], whereas the hyperbolic model has been implemented in C++ based on the fastText code. We apply a minimum count of 15 to discard infrequent words, use a window size of ± 10 words, 10 negative samples and a subsampling factor of 10^{-5}. The vocabulary contains 286013 words. The shift parameter θ was set to 3 for the hyperbolic skip-gram. The distance of the geodesic updates is capped at a maximum of 1 to prevent points moving off of the hyperboloid due to numerical instability. For the hyperbolic model, the two parameter layers are tied and initialized with points sampled from a normal distribution with standard deviation 0.01 around the base point $(0, \dots, 0, 1)$ of the hyperboloid. For fastText, the default initialization scheme is used. Training was run for 3 epochs and the start learning rate was chosen from $\{0.1, 0.05, 0.01, 0.005\}$ and decayed linearly to 0 over the full training time. This learning rate scheme does not guarantee convergence, but has been established to work well empirically. For a theoretical analysis of gradient descent on manifolds and convergence analysis, see e.g. Absil et al. (2008). The experimental results reported are the best achieved for each dimension.

5.2 Implementation details

FastText uses *HogWild* (Niu et al., 2011) as its optimization scheme, i.e. multi-threaded stochastic gradient descent without parameter locking. This allows for embedding vectors being concurrently written by different threads. As the Euclidean optimization is unconstrained, such concurrent writes are unproblematic. In contrast, the hyperbolic optimization is constrained, since the points must always remain on the hyperboloid, and so concurrent writes to an embedding vector could result in an invalid state. For this reason a locking scheme is used to prevent concurrent access to embedding vectors by separate threads. This scheme locks each parameter vector that is currently in-use by a thread (representing the center word, or the context word, or a negative sample) so that no other thread can access it. If a thread can not obtain the locks that it needs for a skip-gram learning task, then this task is skipped.

5.3 Word similarity

The word similarity task consists in rating the level of semantic relatedness between two words and comparing this with human judgment. Concretely, a dataset for similarity evaluation consists of a number of pairs of words, each of which has received a number of ratings by test persons on a predefined scale. From this an average rating is computed for each pair. Given learned word embeddings, the test computes a similarity score for each pair of words and computes the *Spearman rank correlation* ρ between the scores and the annotated ratings.

We evaluate on three different similarity datasets. The WordSimilarity-353 Test Collection (WS-353) is a relatively small dataset of 353 word pairs, that was introduced in Finkelstein et al. (2001). It covers both similarity, i.e. if words are synonyms, and relatedness, i.e. if they appear in the same context. Simlex-999 (Hill et al., 2015) consists of 999 pairs aiming at measuring similarity only, not relatedness or association. Finally,

[1] Available at https://storage.googleapis.com/lateral-datadumps/wikipedia_utf8_filtered_20pageviews.csv.gz

[2] https://github.com/facebookresearch/fastText

the MEN dataset (Bruni et al., 2014) consists of 3000 word pairs covering both similarity and relatedness.

Faruqui and Dyer (2014) introduced evaluation code to compute the average Spearman ρ as a quality measure of the embeddings. For the Euclidean embeddings the cosine similarity is used to rate the similarity of two words. We expand this code for hyperbolic embeddings by using the Minkowski dot product as similarity function, which is anti-monotone to the hyperbolic distance. For each dimension we report the results of the model with the highest weighted average ρ across the three datasets.

The results are shown in Table 1. The hyperbolic skip-gram embeddings give an improved performance for some combinations and datasets. For the WS-353 and MEN datasets, higher scores can mainly be observed in low dimensions (5, 20), whereas for higher dimensions the Euclidean version is superior by a small margin. The relatively low scores on Simlex-999 suggest that both skip-gram models are better at learning relatedness and association. We point out that our results on the WS-353 dataset surpass the ones achieved in Dhingra et al. (2018) by a large margin, which could potentially be due to actually using the hyperbolic distance to compare the similarity of two words.

Overall, we conclude that the proposed method is able to learn sensible embeddings in hyperbolic space and shows potential especially in dimensions that are uncommonly low compared to other algorithms. However, we do not observe the extraordinary performance gains observed for the tree embeddings, where low-dimensional hyperbolic embeddings outperformed Euclidean embeddings by a large margin (Nickel and Kiela, 2017).

5.4 Word analogy

Evaluating word analogy dates back to the seminal word2vec paper (Mikolov et al., 2013). It relates to the idea that the learned word representations exhibit so called *word vector arithmetic*, i.e. semantic and syntactic relationships present themselves as translations in the word vector space. For example the relationship between a country and its capital would be encoded in their difference vector and is approximately the same for different instances of the relation, e.g. $vec(France) - vec(Paris) \approx vec(Germany) - vec(Berlin)$.

Evaluating the extent to which these relations are fulfilled can then serve as a proxy for the quality of the embeddings. The dataset from Mikolov et al. (2013) consists of roughly 20,000 relations in the form $A : B = C : D$, and the task is to find $vec(D)$ as the closest vector to $vec(B) - vec(A) + vec(C)$. As implementation details, all vectors are normalized to unit norm before computing the compound vector, and the three query words are removed from the corpus before computing the nearest neighbor.

Using the analogy task for hyperbolic word embeddings needs some adjustment, since $\mathbb{H}^n$ is not a vector space. Rather, the Riemannian structure has to be used to relate the four embeddings of the relation. Let Log_p the inverse of the exponential map Exp_p. We propose the following procedure as the natural generalization of the analogy task to curved manifolds such as hyperbolic space:

> Let $A : B = C : D$ be the relation to be evaluated and identify the associated word embeddings in $\mathbb{H}^n$ with the same symbols. Then
>
> 1. Compute $w = \mathrm{Log}_A(B)$.
> 2. Compute $v = \mathrm{Log}_A(C)$.
> 3. Parallel transport w along the geodesic connecting A to C, resulting in $\varphi_v(w) \in T_C\mathbb{H}^n$.
> 4. Calculate the point $Z = \mathrm{Exp}_C(\varphi_v(w))$.
> 5. Search for the closest point to Z using the hyperbolic distance.

The result of the first step (corresponding to the vector $B - A$ in the Euclidean formulation), is an element of the tangent space $T_A\mathbb{H}^n$ at A. In order to "add" this vector to C however, it needs to be moved to the tangent space $T_C\mathbb{H}^n$ using parallel transport along the geodesic connecting A and C. Addition in Euclidean space is following a geodesic starting at C in the direction $B - A$. In $\mathbb{H}^n$, this is achieved by following the geodesic along the tangent vector obtained by parallel transport. The resulting point $Z \in \mathbb{H}^n$ can then be used for the usual nearest neighbor search among all words.

This procedure seems indeed to be the natural generalization of the analogy task. There is a seri-

	Euclidean			Hyperbolic		
Dim/Dataset	WS-353	Simlex	MEN	WS-353	Simlex	MEN
5	0.3508	**0.1622**	0.4152	**0.3635**	0.1460	**0.4655**
20	0.5417	0.2291	0.6433	**0.6156**	**0.2554**	**0.6694**
50	0.6628	0.2738	**0.7217**	**0.6787**	**0.2784**	0.7117
100	**0.6986**	**0.2923**	**0.7473**	0.6846	0.2832	0.7217

Table 1: Spearman rank correlation on 3 similarity datasets. Word vectors trained on 498M words from Wikipedia.

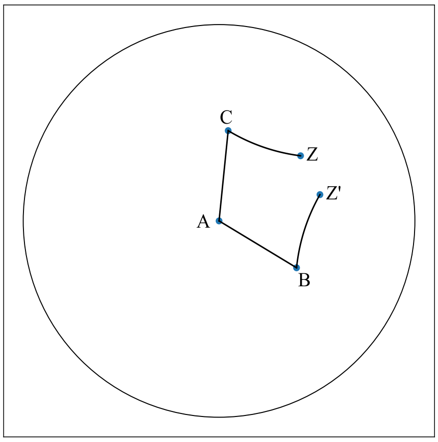

Figure 3: The analogue of the word analogy task on the hyperbolic plane, depicted using the Poincaré disc model. The curved lines are the geodesic line segments connecting the points, and the opposite sides have the equal hyperbolic length. The generalization of the word analogy task results in either of two distinct points Z, Z', depending on the arbitrary choice of going via B, or via C, having started at A.

ous problem, however. The procedure obtains the point Z by beginning at A and proceeding via C, and this point Z is then used to search for nearest neighbors. However, it would have been equally valid to proceed in the opposite sense, i.e. by beginning at A and proceeding via B (parallel transporting the appropriate vector), and this would also yield a point Z'. In Euclidean space, it doesn't matter which of these two alternatives is followed, since the resulting points Z, Z' coincide (indeed, in the Euclidean case the points $A, B, C, Z = Z'$ form a parallelogram).

However, in hyperbolic space, or indeed on any manifold of constant non-zero curvature, the two senses of the procedure yield distinct points, i.e. $Z \neq Z'$. This is a serious problem with the proposed generalization of the word analogy task to curved spaces, since there is a *choice* of points

Dim.	5	20	50	100
Euclidean	0.0011	0.2089	0.3866	0.5513
Hyperbolic	0.0020	0.2251	0.3536	0.3636

Table 2: Accuracy on the Google word analogy dataset. Word vectors trained on 498M words from Wikipedia.

Z, Z' near which we could search, but there is no valid means of deciding which to use. Figure 3 depicts the situation in hyperbolic space for a typical choice of points A, B, C and the resultant points Z, Z'. The Poincaré disc model is used for this illustration.

Table 2 shows the performance on the analogy task of the best embeddings from the word similarity task assessment, where one of the two choices of Z was chosen arbitrarily. In view of the conceptual problems with the generalization, these measurements are included only for the sake of completeness.

6 Conclusions and outlook

We presented a first attempt at learning word embeddings in hyperbolic space from free text input. The hyperbolic skip-gram model compared favorably to its Euclidean counterpart for some common similarity datasets, especially in low dimensions.

We discussed also the problem inherent in the straight-forward generalization of the word analogy task to curved manifolds such as hyperbolic space. Further research is required to reconceptualize the task in the Euclidean case such that it may be applied also to curved manifolds.

A crucial point for further investigation is the formulation of the objective function. The proposed one is only one possible choice of how to use the hyperbolic structure on top of the skip-gram model. Further experiments might be conducted to potentially increase the performance of hyperbolic word embeddings.

Another important direction for future research is the development of the necessary algorithms to use hyperbolic embeddings for downstream tasks. Since many common implementations of classifiers assume Euclidean input data as features, this would require reformulating algorithms so that they can be used in hyperbolic space. For example, Lebanon and Lafferty (2004) serves as an example of how to translate a learning algorithm by interpreting the objective function geometrically and applying the corresponding concepts from Riemannian geometry in case of data on a manifold. In another recent work, Ganea et al. (2018) derive hyperbolic versions of various neural network architectures that can be used on top of hyperbolic embeddings.

Code

The implementation of the hyperbolic skip-gram training and experiments is available online.[3]

References

P.-A. Absil, R. Mahony, and R. Sepulchre. 2008. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Silvère Bonnabel. 2011. Stochastic gradient descent on Riemannian manifolds. *arXiv:1111.5280*.

Elia Bruni, Nam Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *J. Artif. Int. Res.*, 49(1):1–47.

Benjamin P. Chamberlain, James Clough, and Marc P. Deisenroth. 2017. Neural Embeddings of Graphs in Hyperbolic Space. *arXiv:1705.10359*.

Chris De Sa, Albert Gu, Chris Ré, and Fred Sala. 2018. Representation Tradeoffs for Hyperbolic Embeddings. *arXiv:1804.03329*.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

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. Association for Computational Linguistics.

Manaal Faruqui and Chris Dyer. 2014. Community evaluation and exchange of word vectors at wordvectors.org. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL)*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 406–414, New York, NY, USA. ACM.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. *CoRR*, abs/1805.09112.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with genuine similarity estimation. *Comput. Linguist.*, 41(4):665–695.

Guy Lebanon and John Lafferty. 2004. Hyperplane margin classifiers on the multinomial manifold. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, New York, NY, USA. ACM.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Andriy Mnih and Geoffrey Hinton. 2008. A scalable hierarchical distributed language model. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, pages 1081–1088, USA. Curran Associates Inc.

Maximilian 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, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 3776–3785.

Maximillian 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 6338–6347. Curran Associates, Inc.

[3] `https://github.com/lateral/minkowski`

Feng Niu, Benjamin Recht, Christopher Re, and Stephen J. Wright. 2011. HOGWILD!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent. *arXiv:1106.5730*.

Peter Petersen. 2006. *Riemannian Geometry*. Graduate Texts in Mathematics. Springer New York.

William F. Reynolds. 1993. Hyperbolic geometry on a hyperboloid. *The American Mathematical Monthly*, 100(5):442–455.

Benjamin Wilson and Matthias Leimeister. 2018. Gradient descent in hyperbolic space. *arXiv:*.