POINCARÉ GLOVE: HYPERBOLIC WORD EMBEDDINGS

Anonymous authors
Paper under double-blind review

ABSTRACT

Words are not created equal. In fact, they form an aristocratic graph with a latent hierarchical structure that the next generation of unsupervised learned word embeddings should reveal. In this paper, driven by the notion of delta-hyperbolicity or tree-likeliness of a space, we propose to embed words in a Cartesian product of hyperbolic spaces which we theoretically connect with the Gaussian word embeddings and their Fisher distance. We adapt the well-known Glove algorithm to learn unsupervised word embeddings in this type of Riemannian manifolds. We explain how concepts from the Euclidean space such as parallel transport (used to solve analogy tasks) generalize to this new type of geometry. Moreover, we show that our embeddings exhibit hierarchical and hypernymy detection capabilities. We back up our findings with extensive experiments in which we outperform strong and popular baselines on the tasks of similarity, analogy and hypernymy detection.

1 INTRODUCTION & MOTIVATION

Word embeddings are ubiquitous nowadays as first layers in neural network and deep learning models for natural language processing. They are essential in order to move from the discrete word space to the continuous space where differentiable loss functions can be minimized. The popular models of Glove (Pennington et al., 2014), Word2Vec (Mikolov et al., 2013) or FastText (Bojanowski et al., 2016), provide efficient ways to learn words vectors fully unsupervised from raw text corpora, solely based on word co-occurrence statistics. These models can be successfully applied to word similarity and other downstream tasks and, surprisingly (or not (Arora et al., 2016)), exhibit a linear algebraic structure that is also useful to solve word analogy.

However, unsupervised word embeddings still largely suffer from revealing asymmetric word relations including the latent hierarchical structure of words. This is currently one of the key limitations in automatic text understanding, e.g. for tasks such as textual entailment (Bowman et al., 2015). To address this issue, (Vilnis & McCallum, 2015; Muzellec & Cuturi, 2018) propose to move from point embeddings to probability density functions, the simplest being Gaussian or Elliptical distributions. Their intuition is that the variance of such a distribution should encode the generality/specificity of the respective word. However, this method results in losing the arithmetic properties of point embeddings (e.g. for analogy reasoning) and becomes unclear how to properly use them in downstream tasks. To this end, we propose to take the best from both worlds: we embed words as points in a Cartesian product of hyperbolic spaces and, additionally, explain how they are bijectively mapped to Gaussian embeddings with diagonal covariance matrices, where the hyperbolic distance between two points becomes the Fisher distance between the corresponding probability distribution functions (PDFs). We learn these embeddings unsupervised from raw text by generalizing the Glove method. Moreover, the linear arithmetic property used for solving word analogy has a mathematical grounded correspondence in this new space based on the notion of parallel transport. In addition, these hyperbolic embeddings exhibit hierarchical structure useful for hypernymy detection (via their PDFs) and outperform Euclidean Glove on similarity benchmarks. Finally, they can be used as inputs for downstream tasks as explained by the recent work of Ganea et al. (2018b).

We provide additional reasons for choosing the hyperbolic geometry to embed words. We explain the notion of average δ-hyperbolicity of a graph, a geometric quantity that measures its "democracy" (Borassi et al., 2015). A small hyperbolicity constant implies "aristocracy", namely the existence of a small set of nodes that "influence" most of the paths in the graph. It is known that real-world graphs are mainly complex networks (e.g. scale-free exhibiting power-law node degree distributions) which in turn are better embedded in a tree-like space, i.e. hyperbolic (Krioukov et al., 2010). Since,

intuitively, words form an "aristocratic" community (few generic ones defining different topics; many specific ones) and since a big subset of them exhibits a hierarchical structure (e.g. WordNet (Miller et al., 1990)), it is naturally to learn hyperbolic word embeddings. Moreover, we empirically measure very low average δ-hyperbolicity constants of some variants of the word log-co-occurrence graph (used by the Glove method), providing additional quantitative reasons for why spaces of negative curvature (i.e. hyperbolic) are better suited for word representations.

2 RELATED WORK

Recent *supervised methods* can be applied to embed any tree or directed acyclic graph in a low dimensional space with the aim of improving link prediction either by imposing a partial order in the embedding space (Vendrov et al., 2015; Vilnis et al., 2018; Athiwaratkun & Wilson, 2018), by using hyperbolic geometry (Nickel & Kiela, 2017; 2018), or both (Ganea et al., 2018a).

To learn *word embeddings* that exhibit hypernymy or hierarchical information, supervised methods (Vulić & Mrkšić, 2018; Nguyen et al., 2017) leverage external information (e.g. WordNet) together with raw text corpora. However, the same goal is also targeted by more ambitious fully unsupervised models which move away from the "point" assumption and learn various probability densities for each word (Vilnis & McCallum, 2015; Muzellec & Cuturi, 2018; Athiwaratkun & Wilson, 2017; Singh et al., 2018).

There have been two very recent attempts at learning unsupervised word embeddings in the hyperbolic space (Leimeister & Wilson, 2018; Dhingra et al., 2018). However, they suffer from either not being competitive on standard tasks, not showing the benefit of using hyperbolic spaces or not being trained on a realistically large corpus. We address these problems and, moreover, the connection with density based methods is made explicit and leveraged to improve hypernymy detection.

3 HYPERBOLIC SPACES AND THEIR CARTESIAN PRODUCT

In order to work in the hyperbolic space, we have to choose one *model*, among the five isometric models that exist. In this work, the one that we choose to embed words in is the Poincaré ball $\mathbb{D}^n = \{x \in \mathbb{R}^n \mid \|x\|_2 < 1\}$, illustrated in Figure 1a for $n = 2$, where dark lines represent geodesics. The distance function is given by $d_{\mathbb{D}^n}(x, y) = \cosh^{-1}\left(1 + \lambda_x \lambda_y \|x - y\|_2^2 / 2\right)$, $\lambda_x := 2/(1 - \|x\|_2^2)$ being the *conformal factor*. We will also embed words in products of hyperbolic spaces, and explain why later on. A product of p balls $(\mathbb{D}^n)^p$, with the induced product geometry, is known to have distance function $d_{(\mathbb{D}^n)^p}(x, y)^2 = \sum_{i=1}^p d_{\mathbb{D}^n}(x_i, y_i)^2$. Finally, another model of interest for us is the Poincaré half-plane $\mathbb{H}^2 = \mathbb{R} \times \mathbb{R}_+^*$ illustrated in Figure 1d, with distance function $d_{\mathbb{H}^2}(x, y) = \cosh^{-1}\left(1 + \|x - y\|_2^2 / (2y_1 y_2)\right)$. Figure 2 shows an isometry φ from $\mathbb{D}^2$ to $\mathbb{H}^2$ mapping the vertical segment $\{0\} \times (-1, 1)$ to $\mathbb{R}_+^*$ and fixing $(0, 1)$, sending the radius to ∞.

(a) (b) (c) (d)

Figure 1: Isometric deformation φ of $\mathbb{D}^2$ into $\mathbb{H}^2$.

4 ADAPTING GLOVE

Euclidean GLOVE. The original GLOVE (Pennington et al., 2014) algorithm is an unsupervised method for learning word representations in a Euclidean space from statistics of word co-occurences in a corpus of texts, with the aim to capture some of the words' meaning and relations geometrically.

We use standard notations: X_{ij} is the number of times word j occurs in the context of word i; $X_i = \sum_k X_{ik}$; $P_{ij} = X_{ij}/X_i$ is the probability that word j appears in the context of word i. An embedding of a (target) word i is written w_i, while an embedding of a context word k is written $\tilde{w}_k$.

The initial formulation of the GLOVE model suggests to learn embeddings as to satisfy the equation $w_i^T \tilde{w}_k = \log(P_{ik}) = \log(X_{ik}) - \log(X_i)$. Since X_{ik} is symmetric in (i, k) but P_{ik} is not, they

propose to restore the symmetry by introducing biases for each word, absorbing $\log(X_i)$ into i's bias:

$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = \log(X_{ik}). \tag{1}$$

Finally, the authors suggest to enforce this equality by optimizing a weighted least-square loss:

$$J = \sum_{i,j=1}^{V} f(X_{ij}) \left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij} \right)^2, \tag{2}$$

where V is the size of the vocabulary and f downweights the signal coming from frequent words (it is typically chosen to be $f(x) = \min\{1, (x/x_m)^\alpha\}$, with $\alpha = 3/4$ and $x_m = 100$).

GLOVE in metric spaces. Note that there is no clear correspondence of the Euclidean inner-product in a hyperbolic space. However, we are provided with a distance function. Further notice that one could rewrite Eq. (1) with the Euclidean distance as $-\frac{1}{2}\|w_i - \tilde{w}_k\|^2 + b_i + \tilde{b}_k = \log(X_{ik})$, where we absorbed the squared norms of the embeddings into the biases. We thus replace the GLOVE loss by:

$$J = \sum_{i,j=1}^{V} f(X_{ij}) \left(-h(d(w_i, \tilde{w}_j)) + b_i + \tilde{b}_j - \log X_{ij} \right)^2, \tag{3}$$

where h is a function to be chosen as a hyperparameter of the model, and d can be any differentiable distance function. Although the most direct correspondence with GLOVE would suggest $h(x) = x^2/2$, we sometimes obtained better results with other functions, such as $h = \cosh^2$ (see sections 7 & 8). Note that De Sa et al. (2018) also apply $\cosh$ to their distance matrix for hyperbolic MDS before applying PCA. Understanding why $h = \cosh^2$ is a good choice would be interesting future work.

5 CONNECTING GAUSSIAN EMBEDDINGS & HYPERBOLIC EMBEDDINGS

In order to endow Euclidean word embeddings with richer information, Vilnis & McCallum (2015) proposed to represent words as Gaussians, *i.e.* with a mean vector and a covariance matrix[1], expecting the variance parameters to capture how generic/specific a word is, and hopefully entailment relations. On the other hand, Nickel & Kiela (2017) proposed to embed words of the WordNet hierarchy (Miller et al., 1990) in hyperbolic space, because this space is mathematically known to be better suited to embed tree-like metric spaces. It is hence natural to wonder: is there a connection between the two?

The Fisher geometry of Gaussians is hyperbolic. It turns out that there exists a striking connection (Costa et al., 2015). Note that a *1D Gaussian* $\mathcal{N}(\mu, \sigma^2)$ *can be represented as a point* (μ, σ) in $\mathbb{R} \times \mathbb{R}_+^*$. Then, the Fisher distance between two distributions relates to the hyperbolic distance in $\mathbb{H}^2$:

$$d_F\left(\mathcal{N}(\mu, \sigma^2), \mathcal{N}(\mu', \sigma'^2)\right) = \sqrt{2} d_{\mathbb{H}^2}\left((\mu/\sqrt{2}, \sigma), (\mu'/\sqrt{2}, \sigma')\right). \tag{4}$$

For n-dimensional Gaussians with diagonal covariance matrices written $\Sigma = \mathrm{diag}(\sigma)^2$, it becomes:

$$d_F\left(\mathcal{N}(\mu, \Sigma), \mathcal{N}(\mu', \Sigma')\right) = \sqrt{\sum_{i=1}^{n} 2 d_{\mathbb{H}^2}\left((\mu_i/\sqrt{2}, \sigma_i), (\mu_i'/\sqrt{2}, \sigma_i')\right)^2}. \tag{5}$$

Hence there is a direct correspondence between diagonal Gaussians and the product space $(\mathbb{H}^2)^n$.

Fisher distance, KL & Gaussian embeddings. The above paragraph lets us relate the WORD2GAUSS algorithm (Vilnis & McCallum, 2015) to hyperbolic word embeddings. Although one could object that WORD2GAUSS is trained using a KL divergence, while hyperbolic embeddings relate to Gaussian distributions via the Fisher distance d_F, let us remind that KL and d_F define the same local geometry. Indeed, the KL is known to be related to d_F, as its local approximation (Jeffreys, 1946). In short, if $P(\theta + d\theta)$ and $P(\theta)$ denote two closeby probability distributions for a small $d\theta$, then $\mathrm{KL}(P(\theta + d\theta)||P(\theta)) = (1/2)\sum_{ij} g_{ij} d\theta^i d\theta^j + \mathcal{O}(\|d\theta\|^3)$, where $(g_{ij})_{ij}$ is the Fisher information metric, inducing d_F.

[1] diagonal or even spherical, for simplicity.

Riemannian optimization. A benefit of representing words in (products of) hyperbolic spaces, as opposed to (diagonal) Gaussian distributions, is that one can use recent Riemannian adaptive optimization tools such as RADAGRAD (Bécigneul & Ganea, 2018). Note that without this connection, it would be unclear how to define a variant of ADAGRAD (Duchi et al., 2011) intrinsic to the statistical manifold of Gaussians. Empirically, we indeed noticed better results using RADAGRAD, rather than simply Riemannian SGD (Bonnabel, 2013). Similarly, note that GLOVE trains with ADAGRAD.

6 ANALOGIES AND ENTAILMENT FOR HYPERBOLIC/GAUSSIAN EMBEDDINGS

The connection exposed in section 5 allows us to provide mathematically grounded *(i)* analogy computations for Gaussian embeddings using hyperbolic geometry, and *(ii)* hypernymy detection for hyperbolic embeddings using Gaussian distributions.

6.1 COMPUTING ANALOGIES

A common task used to evaluate word embeddings, called *analogy*, consists in finding which d is to c, what b is to a. For instance, *queen* is to *woman* what *king* is to *man*. In the Euclidean space, the solution to this problem is usually taken as $d = c + (b - a) = b + (c - a)$, which is also to b, what c is to a. But how should one intrinsically define "analogy parallelograms" in a space of Gaussian distributions? Note that Vilnis & McCallum (2015) do not evaluate their Gaussian embeddings on the analogy task, and that it would be unclear how to do so. However, since we can go back and forth between (diagonal) Gaussians and (products of) hyperbolic spaces as explained in section 5, we can use the fact that parallelograms are naturally defined in the Poincaré ball, by the notion of gyro-translation (Ungar, 2012, section 4). In the Poincaré ball, the two solutions $d_1 = c + (b - a)$ and $d_2 = b + (c - a)$ are respectively generalized to

$$d_1 = c \oplus gyr[c, \ominus a](\ominus a \oplus b), \quad \text{and} \quad d_2 = b \oplus gyr[b, \ominus a](\ominus a \oplus c). \tag{6}$$

The formulas for these operations are described in closed-forms in appendix C, and are easy to implement. The fact that these two differ is due to the curvature of the space. For evaluation, we chose the gyro-midpoint between these two solutions, defined as $m_{d_1 d_2} := d_1 \oplus ((-d_1 \oplus d_2) \otimes (1/2)) = m_{d_2 d_1}$, which is the point located on the geodesic between d_1 and d_2, at equal hyperbolic distance between from d_1 as from d_2. Note that one can rewrite Eq. (6) with tools from differential geometry as

$$c \oplus gyr[c, \ominus a](\ominus a \oplus b) = \exp_c(P_{a \to c}(\log_a(b))), \tag{7}$$

where $P_{x \to y} = (\lambda_x / \lambda_y) gyr[y, \ominus x]$ denotes the *parallel transport* along the unique geodesic from x to y. The exp and log maps of Riemannian geometry were related to the theory of gyrovector spaces (Ungar, 2008) by Ganea et al. (2018b), who also mention that when continuously deforming the hyperbolic space $\mathbb{D}^n$ into the Euclidean space $\mathbb{R}^n$, sending the curvature from -1 to 0 (*i.e.* the radius of $\mathbb{D}^n$ from 1 to ∞), the Möbius operations $\oplus^\kappa, \ominus^\kappa, \otimes^\kappa, gyr^\kappa$ recover their respective Euclidean counterparts $+, -, \cdot, Id$. Hence, the analogy solutions $d_1, d_2, m_{d_1 d_2}$ of Eq. (6) would then all recover $d = c + b - a$, which seems a nice sanity check.

6.2 A SCORE FOR ENTAILMENT/HYPERNYMY

Poincaré embeddings. Nickel & Kiela (2017) use a heuristic entailment score in order to predict whether u *is a* v, defined for $u, v \in \mathbb{D}^n$ as is-a$(u, v) := -(1 + \alpha(\|v\|_2 - \|u\|_2))d(u, v)$, where $\alpha = 10^3$, based on the intuition that higher concepts in the hierarchy should be embedded with a lower Euclidean norm. However, such a choice *is not intrinsic* to the hyperbolic space. The loss that they use for training only involves the distance function, hence training is intrinsic to $\mathbb{D}^n$, *i.e.* invariant to applying any isometry $\varphi : \mathbb{D}^n \to \mathbb{D}^n$ to all word embeddings, but their "is-a" score is not.

The importance of fixing the isometry. We argue that generality of a concept as being embedded in a Poincaré ball is better captured by a *direction* in hyperbolic space, *i.e.* by a geodesic, rather than by the distance from the origin. Why? For a 1D Gaussian $\mathcal{N}(\mu, \sigma^2)$ representing a concept, generality can be nicely encoded in the magnitude of σ. As shown in section 5, we can map the space of Gaussians endorsed with the Fisher distance to the hyperbolic upper half-plane $\mathbb{H}^2$, where σ

corresponds to the (positive) second coordinate in $\mathbb{H}^2 = \mathbb{R} \times \mathbb{R}^*_+$. Moreover, as shown in section 3, $\mathbb{H}^2$ can be isometrically mapped to $\mathbb{D}^2$, where the second coordinate $\sigma \in \mathbb{R}^*_+$ corresponds to the open vertical segment $\{0\} \times (-1, 1)$ in $\mathbb{D}^2$. However, in $\mathbb{D}^2$, any rotation w.r.t. the origin or any (hyperbolic) translation is an isometry[2]. Hence, in order to map a word $x \in \mathbb{D}^2$ to a Gaussian $\mathcal{N}(\mu, \sigma^2)$ via $\mathbb{H}^2$, we first have to align $\{0\} \times (-1, 1)$ with whichever geodesic in $\mathbb{D}^2$ encodes generality by applying the correct isometry to $\mathbb{D}^2$ (centering, rotation). These three steps are illustrated in Figure 2.

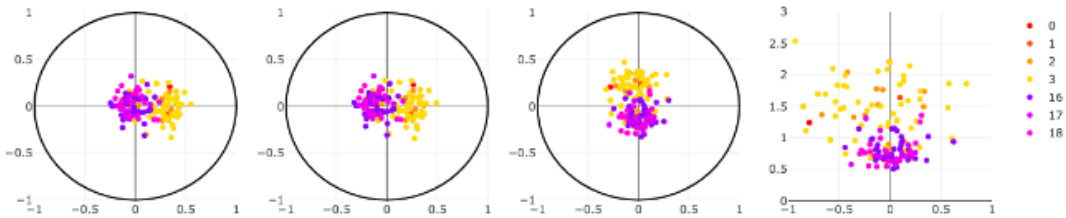

Figure 2: We show here one of the $\mathbb{D}^2$ of some 20D embeddings trained unsupervised with our hyperbolic GLOVE in $(\mathbb{D}^2)^{10}$. These images illustrate the three steps of applying the isometry. From left to right: the trained embeddings; after centering; after rotation; after isometrically mapping them to $\mathbb{H}^2$ as explained in section 3. The isometry was obtained with the semi-supervised method described below, using 400 top and 400 bottom words from WordNet. Experimental details: $h(x) = x^2$; unrestricted vocabulary of 180k words; learning-rate of 0.05.

Gaussian embeddings. Vilnis & McCallum (2015) propose using is-a$(P, Q) := -KL(P||Q)$ for distributions P, Q, the argument being that a low $KL(P||Q)$ indicates that we can encode Q easily as P, implying that Q entails P. However, we would like to mitigate this statement. Indeed, if $P = \mathcal{N}(\mu, \sigma)$ and $Q = \mathcal{N}(\mu, \sigma')$ are two 1D Gaussian distributions with same mean, then $KL(P||Q) = z^2 - 1 - \log(z)$ where $z := \sigma/\sigma'$, which is not a monotonic function of z. *This breaks the idea that the magnitude of the variance should encode the generality/specificity of the concept.*

Another entailment score for Gaussian embeddings. What would constitute a good number for the variance's magnitude of a n-dimensional Gaussian distribution $\mathcal{N}(\mu, \Sigma)$? It is known that 95% of its mass is contained within a hyperellipsoid of volume $V_\Sigma = V_n \sqrt{\det(\Sigma)}$, where V_n denotes the volume of a ball of radius 1 in $\mathbb{R}^n$. For simplicity, we propose dropping the dependence in μ and define a simple score is-a$(\Sigma, \Sigma') := \log(V'_\Sigma) - \log(V_\Sigma) = \sum_{i=1}^n (\log(\sigma'_i) - \log(\sigma_i))$. Note that using difference of logarithms has the benefit of removing the scaling constant V_n, and makes the entailment score invariant to a rescaling of the covariance matrices: is-a$(r\Sigma, r\Sigma') =$ is-a(Σ, Σ').

Computing the score for hyperbolic embeddings. To compute our is-a score between two hyperbolic embeddings, we first map them to Gaussians and then apply the above proposed is-a score. However, as previously explained, in order to map a point $x \in (\mathbb{D}^n)^p$ to a diagonal gaussian $\mathcal{N}(\mu, \Sigma)$, one first has to fix the correct isometry. To do so, we start by identifying two sets G and S of potentially generic and specific words, respectively. For the re-centering, we then compute the means g and s of G and S respectively, and $m := (s + g)/2$, and Möbius translate all words by the global mean with the operation $w \mapsto \ominus m \oplus w$. For the rotation, we set $u := (\ominus m \oplus g)/\| \ominus m \oplus g\|_2$, and rotate all words so that u is mapped to $(0, 1)$. Figure 2 illustrates these steps. In order to identify the two sets G and S, we propose *(i)* a fully unsupervised method, and *(ii)* a semi-supervised one, using a (varying) number of words from the WordNet hierarchy. In *(i)*, we first define a restricted vocabulary of the 50k most frequent words among the unrestricted one of 200k words, and rank them by frequency; we then define G as the 5k most frequent ones, and S as the 5k least frequent ones of the 50k vocabulary (to avoid extremely rare words which might have received less signal during training). In *(ii)*, we define G as k top words from the top 4 levels of the WordNet hierarchy, and S as k bottom words from the bottom 3 levels. Results are described in section 8. As can be seen, the semi-supervised method works better, but using only $k = 20$ top and bottom words $-$*i.e.* a very small amount of supervision $-$ already gives good improvements.

[2]See `http://bulatov.org/math/1001` for intuitive animations describing hyperbolic isometries.

h(x)	d_{avg}	δ_{avg}	$2\delta_{avg}/d_{avg}$
$\log(x^2)$	18950.474	8498.6474	0.8969
x	18.9505	0.7685	0.0811
x^2	4.3465	0.088	0.0405
$\cosh(x)$	3.68	0.0384	0.0209
$\cosh^2(x)$	2.3596	0.0167	0.0142
$\cosh^3(x)$	1.9716	0.0102	0.0103
$\cosh^4(x)$	1.7918	0.0072	0.0081
$\cosh^5(x)$	1.6888	0.0056	0.0066
$\cosh^{10}(x)$	1.4947	0.0026	0.0034

Table 1: δ-hyperbolicity, average distance and their ratio computed via sampling for the metrics given by different h functions.

7 EMBEDDING SYMBOLIC DATA IN A CONTINUOUS SPACE WITH MATCHING HYPERBOLICITY

Why would we embed words in a hyperbolic space? Given some symbolic data, such as a vocabulary along with similarity measures between words $-$ in our case, co-occurence counts X_{ij} $-$ can we understand in a principled manner which geometry would represent it best? Choosing the right metric space to embed words can be understood as selecting the right inductive bias $-$ an essential step.

δ-hyperbolicity. A particular quantity of interest describing qualitative aspects of metric spaces is the *δ-hyperbolicity*, introduced by Gromov (1987). However, for various reasons discussed in appendix B, we used the *averaged* δ-hyperbolicity, denoted δ_{avg}, defined by Albert et al. (2014). Intuitively, a low δ_{avg} of a finite metric space characterizes that this space has an underlying hyperbolic geometry, *i.e.* an approximate tree-like structure, and that the hyperbolic space would be well suited to isometrically embed it. We also report the ratio $2 * \delta_{avg}/d_{avg}$ (invariant to metric scaling), where d_{avg} is the average distance in the finite space, as suggested by Borassi et al. (2015), whose low value also characterizes the "hyperbolicness" of the space.

Computing δ_{avg}. Since our methods are trained on a weighted graph of co-occurences, we want to compute its hyperbolicity δ_{avg}. However, in order to do so, one needs to be provided with a distance $d(i, j)$ for each pair of words (i, j), while our symbolic data is only made of similarities measures. Note that we cannot associate the value $-\log(P_{ij})$ to $d(i, j)$, as this quantity is not symmetric. Instead, inspired from Eq. (3), we associate to words i, j the distance[3] $h(d(i, j)) := -\log(X_{ij}) + b_i + b_j \geq 0$ with the choice $b_i := \log(X_i)$, *i.e.* $d(i, j) = h^{-1}(\log((X_i X_j)/X_{ij}))$.

Table 1 shows values for different choices of h. The discrete metric spaces we obtained for our symbolic data of co-occurences appear to have a very low hyperbolicity, *i.e.* to be very much "hyperbolic", which suggests to embed words in (products of) hyperbolic spaces. We report in section 8 empirical results for $h(x) = x^2$ and $h(x) = \cosh(x)$.

8 EXPERIMENTS: SIMILARITY, ANALOGY, ENTAILMENT

We trained all models on a corpus provided by Omer Levy and which was also used to report the results on some works on word embeddings. The preprocessing of the corpus is explained in detail in these works (Levy & Goldberg, 2014; Levy et al., 2015). The dataset has been obtained from an English Wikipedia dump and contains 1.4 billion tokens. Words that appear less than 100 times in the corpus have been discarded leaving 189,533 unique tokens. The co-occurrence matrix contains 700 million non-zero entries, for a symmetric window size of 10.

We report results for both 100D embeddings trained in a 100D Poincaré ball, and for 50x2D embeddings, which were trained in the Cartesian product of 50 2D Poincaré balls. Note that in the

[3]One can replace $\log(x)$ with $\log(1 + x)$ to avoid computing the logarithm of zero.

case of both models, one word will be represented by a 100D array. For the Poincaré models we employ both $h(x) = x^2$ and $h(x) = \cosh^2(x)$.

We compare against a 100D Euclidean GloVe model which was trained using the hyperparameters suggested in the work that introduced GloVe.

All models were trained for 50 epochs, on the full corpus of 189,533 word types. The vanilla GloVe model has been trained using AdaGrad. All hyperbolic models have been trained using RAdaGrad. For the Euclidean baseline as well as for models with $h(x) = x^2$ we used a learning rate of 0.05. For Poincaré models with $h(x) = \cosh^2(x)$ we used a learning rate of 0.01. All models have biases.

For another experiment setting we initialize our Poincaré model with pretrained parameters. These were obtained by training the same model and only accounting for the most frequent 50,000 word (instead of considering the full 189,533. This will be referred to as the "initialization trick". For fairness, we also trained the vanilla (Euclidean) GloVe model in the same fashion.

Word similarity is assessed using a number of well established benchmarks shown in the Table 2.

Experiment name	Rare Word	WordSim	SimLex	SimVerb	MC	RG
100D Vanilla GloVe	0.3798	0.5901	0.2963	0.1675	0.6524	0.6894
100D Vanilla GloVe w/ init trick	0.3787	0.5668	0.2964	0.1639	0.6562	0.6757
100D Poincaré GloVe $h(x) = \cosh^2(x)$ w/ init trick	**0.4187**	0.6209	**0.3208**	**0.1915**	**0.7833**	**0.7578**
50x2D Poincaré GloVe $h(x) = \cosh^2(x)$	0.4111	**0.6367**	0.3084	0.1811	0.7748	0.725
50x2D Poincaré GloVe $h(x) = x^2$	0.4106	0.5844	0.3007	0.1725	0.7586	0.7236

Table 2: Word similarity results for 100-dimensional models.

For word analogy, we evaluate on the Google benchmark introduced by Mikolov and its two splits that contain semantic and syntactic analogy queries. We also use a benchmark by MSR that is also commonly employed in other word embedding works. For the Euclidean baselines we use 3COSADD. After applying hyperbolic parallel transport, the best candidate is selected to be the closest word to this point in terms of either the cosine distance, or the Poincaré distance.

Experiment name	Method	SemGoogle	SynGoogle	Google	MSR
100D Vanilla GloVe	3COSADD	0.6005	0.5869	0.5931	0.4868
100D Vanilla GloVe w/ init trick	3COSADD	0.6427	0.595	0.6167	0.4826
100D Poincaré GloVe $h(x) = \cosh^2(x)$, w/ init. trick	Cosine dist	**0.6641**	**0.6209**	**0.6339**	**0.518**
50x2D Poincaré GloVe $h(x) = x^2$	Poincaré dist	0.644	0.5967	0.6182	0.4587

Table 3: Word analogy results for 100-dimensional models.

sixties	seventies, eighties, nineties, 60s, 70s, 60s, 1960s, 80s, 90s, 70s
dance	dancing, dances, music, singing, musical, performing, hip-hop, pop, folk, dancers
daughter	wife, married, mother, cousin, son, niece, granddaughter, husband, sister, eldest
vapor	vapour, refrigerant, liquid, condenses, supercooled, fluid, gaseous, gases, droplet, ammonia
ronaldo	cristiano, ronaldinho, rivaldo, messi, zidane, romario, pele, zinedine, xavi, robinho
mechanic	electrician, fireman, machinist, welder, technician, builder, janitor, trainer, brakeman, fitter
algebra	algebras, homological, heyting, geometry, subalgebra, quaternion, calculus, mathematics, unital, algebraic

Table 4: Nearest neighbors for some words for a 100D hyperbolic embedding model. Poincaré distance has been used for selecting neighbors.

reptile	amphibians, carnivore, crocodilian, fish-like, dinosaur, spinosaurus, alligator, triceratops
food	meal, lunch, breakfast, coffee, rice, seafood, fish
music	performance, composition, contemporary, vocal, folk, rock, jazz, electroacoustic
sport	competition, challenge, fitness, outdoor, cycling, tennis, pankration

Table 5: Some words selected from the 100 nearest neighbors and ordered according to the **I** hypernymy score function for a 50x2D hyperbolic embedding model using $h(x) = x^2$.

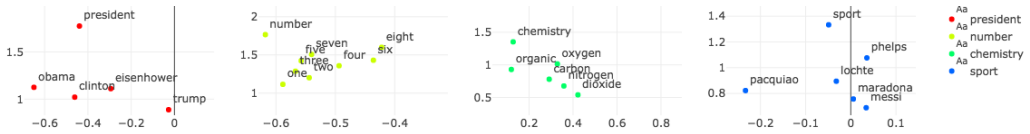

Figure 3: Hierarchies for some concepts captured by a 10x2D model trained with $h(x) = x^2$. Note how different 2D spaces preserve different semantic features better than others.

9 Conclusion and Future Work

We propose to adapt the GloVe algorithm to hyperbolic spaces, and to leberage a connection between statistical manifolds of Gaussian distributions and hyperbolic geometry, in order to better interpret entailment relations between hyperbolic embeddings. We justify the choice of products of hyperbolic spaces as an embedding space via this connection to Gaussian distributions, and via computations of the hyperbolicity of the symbolic data on which GloVe is based. Future work include running experiments to the hyperboloid model, which has been reported to lead to lesser numerical instabilities.

References

Réka Albert, Bhaskar DasGupta, and Nasim Mobasheri. Topological implications of negative curvature for biological and social networks. *Physical Review E*, 89(3):032811, 2014.

Supervised methods	(Kiela et al., 2015)	0.75
	(Weeds et al., 2014)	0.75
	(Nguyen et al., 2017)	0.87
Unsupervised methods	SGNS from (Nguyen et al., 2017)	0.48
Our models		
50x2D, Poincare GloVe, f(dist)=cosh-dist-sq, vocab=180k	WordNet 20+20	0.749
	WordNet 400+400	0.761
	Unsupervised 5k+5k	0.574
50x2D, Poincare GloVe, f(dist)=dist-sq, vocab=180k	WordNet 20+20	0.771
	WordNet 400+400	0.76
	Unsupervised 5k+5k	0.647
50x2D, Poincare GloVe, f(dist)=cosh-dist-sq, vocab=50k	WordNet 20+20	0.739
	WordNet 400+400	0.813
	Unsupervised 5k+5k	0.630
50x2D, Poincare GloVe, f(dist)=dist-sq, vocab=50k	WordNet 20+20	0.752
	WordNet 400+400	0.810
	Unsupervised 5k+5k	0.649

Table 6: WBLESS results

Supervised methods	OrderEmb	0.191
	PARAGRAM + CF	0.32
	WN-Basic	0.24
	WN-WuP	0.214
	WN-LCh	0.214
	WN-Eucl from (Nickel & Kiela, 2017)	0.389
	WN-Poincare	0.512
Unsupervised methods	Word2Gauss-DistPos	0.206
	SGNS-Deps	0.205
	Frequency	0.279
	SLQS-Slim	0.229
	Vis-ID	0.253
Our models (supervised and unsupervised)		
50x2D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, vocab=180k	WordNet 20+20	0.359
	WordNet 400+400	0.411
	Unsupervised 5k+5k	0.277
50x2D, Poincare GloVe, f(dist)=dist-sq, 50ep, vocab=180k	WordNet 20+20	0.365
	WordNet 400+400	0.413
	Unsupervised 5k+5k	0.310
50x2D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, vocab=50k	WordNet 20+20	0.368
	WordNet 400+400	0.412
	Unsupervised 5k+5k	0.295
50x2D, Poincare GloVe, f(dist)=dist-sq, 50ep, vocab=50k	WordNet 20+20	0.365
	WordNet 400+400	0.430
	Unsupervised 5k+5k	0.309

Table 7: Hyperlex results. Baseline results are mostly taken from (Nickel & Kiela, 2017; Vulić et al., 2017)

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399, 2016.

Ben Athiwaratkun and Andrew Wilson. Multimodal word distributions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 1645–1656, 2017.

Ben Athiwaratkun and Andrew Gordon Wilson. Hierarchical density order embeddings. *arXiv preprint arXiv:1804.09843*, 2018.

Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. *arXiv preprint*, 2018.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

Michele Borassi, Alessandro Chessa, and Guido Caldarelli. Hyperbolicity measures democracy in real-world networks. *Physical Review E*, 92(3):032812, 2015.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, 2015.

Wei Chen, Wenjie Fang, Guangda Hu, and Michael W Mahoney. On the hyperbolicity of small-world and treelike random graphs. *Internet Mathematics*, 9(4):434–491, 2013.

Sueli IR Costa, Sandra A Santos, and João E Strapasson. Fisher information distance: a geometrical reading. *Discrete Applied Mathematics*, 197:59–69, 2015. URL `https://arxiv.org/pdf/1210.2354.pdf`.

Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. Representation tradeoffs for hyperbolic embeddings. In *International Conference on Machine Learning*, 2018.

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

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic entailment cones for learning hierarchical embeddings. In *International Conference on Machine Learning*, 2018a.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In *Advances in Neural Information Processing Systems*, 2018b.

Mikhael Gromov. Hyperbolic groups. In *Essays in group theory*, pp. 75–263. Springer, 1987.

Harold Jeffreys. An invariant form for the prior probability in estimation problems. *Proc. R. Soc. Lond. A*, 186(1007):453–461, 1946.

Douwe Kiela, Laura Rimell, Ivan Vulić, and Stephen Clark. Exploiting image generality for lexical entailment detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pp. 119–124, 2015.

Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010.

Matthias Leimeister and Benjamin J Wilson. Skip-gram word embeddings in hyperbolic space. *arXiv preprint arXiv:1809.01498*, 2018.

Omer Levy and Yoav Goldberg. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pp. 171–180. Association for Computational Linguistics, 2014. doi: 10.3115/v1/W14-1618. URL http://www.aclweb.org/anthology/W14-1618.

Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225, 2015. ISSN 2307-387X. URL https://transacl.org/ojs/index.php/tacl/article/view/570.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4): 235–244, 1990.

Boris Muzellec and Marco Cuturi. Generalizing point embeddings using the wasserstein space of elliptical distributions. *arXiv preprint arXiv:1805.07594*, 2018.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. Hierarchical embeddings for hypernymy detection and directionality. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 233–243, 2017.

Maximilian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, 2018.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems*, pp. 6341–6350, 2017.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pp. 1532–43, 2014.

Sidak Pal Singh, Andreas Hug, Aymeric Dieuleveut, and Martin Jaggi. Wasserstein is all you need. *arXiv preprint arXiv:1808.09663*, 2018.

Abraham A Ungar. *Beyond the Einstein addition law and its gyroscopic Thomas precession: The theory of gyrogroups and gyrovector spaces*, volume 117. Springer Science & Business Media, 2012.

Abraham Albert Ungar. A gyrovector space approach to hyperbolic geometry. *Synthesis Lectures on Mathematics and Statistics*, 1(1):1–194, 2008.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. *arXiv preprint arXiv:1511.06361*, 2015.

Luke Vilnis and Andrew McCallum. Word representations via gaussian embedding. *ICLR*, 2015.

Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. Probabilistic embedding of knowledge graphs with box lattice measures. *arXiv preprint arXiv:1805.06627*, 2018.

Ivan Vulić and Nikola Mrkšić. Specialising word vectors for lexical entailment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pp. 1134–1145, 2018.

Ivan Vulić, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. Hyperlex: A large-scale evaluation of graded lexical entailment. *Computational Linguistics*, 43(4):781–835, 2017.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 2249–2259. Dublin City University and Association for Computational Linguistics, 2014.

A MORE EXPERIMENTS IN LOWER DIMENSIONS

We show here extensive results in lower dimensions. Throughout this section, w denotes the target vector, c denotes the context vector, cosh-dist-sq and dist-sq denote using $h(x) = \cosh(x)^2$ and $h(x) = x^2$ respectively; 50ep denotes training with 50 epochs; MIX-Poincare means that we used products of hyperbolic spaces; Vanilla GloVe designate the baseline; For analogy, we use either Poincare distance or cosine similarity to find the nearest neighbor after a gyro-parallel-transport. For analogy, Poincare distance or cosine similarity were used to compute the similarity score.

1	Experiment name		Rare Word (Spearman corr.) using w/w+c	WordSim (Spearman corr.) using w/w+c	SimLex (Spearman corr.) using w/w+c	SimVerb (Spearman corr.) using w/w+c	MC (Spearman corr.) using w/w+c	RG (Spearman corr.) using w/w+c
46	**48D Glove embedding results**							
47	48D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	0.3497 / 0.3467	0.5426 / 0.5444	0.2525 / 0.2499	0.1461 / 0.1404	0.6213 / 0.6540	0.6002 / 0.6192
48		3COSMUL						
49	48D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.3661 / 0.3612	**0.6040** / 0.5802	0.2661 / 0.2652	0.1539 / 0.1598	0.7067 / 0.7172	0.6466 / 0.6557
50		cosine distance	0.2717 / 0.2709	0.5137 / 0.5123	0.2285 / 0.2222	0.1259 / 0.1205	0.5592 / 0.5587	0.5773 / 0.5827
51	48D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	0.3574 / 0.3602	0.4751 / 0.4768	0.2418 / 0.2482	0.1451 / 0.1712	0.6780 / 0.7348	0.6406 / 0.6572
52		cosine distance	0.2423 / 0.2406	0.4952 / 0.4947	0.2341 / 0.2264	0.1192 / 0.1167	0.5688 / 0.5670	0.5393 / 0.5560
55	24x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.3733 / 0.3699	**0.6020** / 0.5976	**0.2678** / **0.2678**	0.1513 / 0.1581	0.7210 / 0.7441	0.6595 / 0.6750
56		cosine distance	0.2640 / 0.2680	0.4614 / 0.4596	0.2068 / 0.1947	0.1119 / 0.1084	0.5792 / 0.6162	0.5768 / 0.5722
57	24x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	**0.3825** / 0.3800	0.5636 / 0.5513	0.2655 / 0.2683	0.1536 / **0.1640**	**0.7857** / 0.7561	0.6959 / **0.7025**
58		cosine distance	0.3006 / 0.2840	0.4726 / 0.4711	0.2261 / 0.2373	0.1121 / 0.1088	0.5027 / 0.5679	0.5201 / 0.4752
59								
60	**20D Glove embedding results**							
61	20D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	0.2930 / 0.2967	0.4412 / 0.4453	0.2153 / 0.2135	0.1153 / 0.1177	0.5120 / 0.5461	0.5367 / 0.5420
62		3COSMUL						
63	20D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.3139 / 0.3089	0.4672 / 0.4605	0.2147 / 0.2200	0.1226 / 0.1295	0.5372 / 0.5685	0.5720 / 0.5753
64		cosine distance	0.2356 / 0.2393	0.3923 / 0.3926	0.1839 / 0.1851	0.0979 / 0.1009	0.4591 / 0.4969	0.4995 / 0.5115
65	20D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	0.3250 / 0.3296	0.4227 / 0.4094	0.1994 / 0.2117	0.1182 / 0.1376	0.5970 / 0.6297	0.6188 / 0.6129
66		cosine distance	0.2279 / 0.2288	0.4365 / 0.4314	0.2021 / 0.1975	0.0951 / 0.0969	0.5507 / 0.5552	0.5105 / 0.5182
69	10x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.3239 / 0.3195	**0.4818** / 0.4771	**0.2329** / **0.2315**	0.1281 / 0.1314	0.6028 / 0.6117	0.5986 / 0.6157
70		cosine distance	0.2140 / 0.2066	0.2705 / 0.2924	0.1741 / 0.1601	0.0963 / 0.0887	0.2069 / 0.2822	0.4495 / 0.4644
71	10x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	**0.3380** / **0.3383**	0.4785 / 0.4706	**0.2314** / 0.2323	0.1239 / **0.1361**	0.6242 / **0.6382**	0.6349 / **0.6375**
72		cosine distance	0.2401 / 0.2183	0.3252 / 0.3112	0.1585 / 0.1560	0.0570 / 0.0628	0.2530 / 0.3422	0.4667 / 0.4842
73								
74	**4D Glove embedding results**							
75	4D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	0.1445 / 0.1440	0.1947 / 0.2009	0.1356 / 0.1319	0.0602 / **0.0628**	0.2701 / 0.2688	0.3403 / 0.3412
76		3COSMUL						
77	4D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.1901 / 0.1884	0.2424 / 0.2430	**0.1432** / 0.1364	0.0581 / 0.0528	0.2299 / 0.2239	0.3065 / 0.2952
78		cosine distance	0.1198 / 0.1160	0.1306 / 0.1414	0.1217 / 0.1175	0.0539 / 0.0477	0.2207 / 0.1914	0.3333 / 0.3363
79	4D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	**0.2031** / 0.2016	0.2782 / **0.2802**	0.1395 / 0.1336	**0.0612** / 0.0577	**0.3173** / **0.3146**	**0.3626** / **0.3627**
80		cosine distance	0.1316 / 0.1327	0.2002 / 0.2009	0.1187 / 0.1178	0.0480 / 0.0459	0.2637 / 0.2441	0.3532 / 0.3598

Figure 4: Full 180K vocabulary, similarity results.

	A	B	Q	R	S	T
1	**Experiment name**		**Semantic Google analogy accuracy using w/w+c**	**Syntactic Google analogy accuracy using w/w+c**	**Total Google analogy accuracy using w/w+c**	**MSR analogy accuracy using w/w+c**
46	**48D Glove embedding results**					
47	48D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	0.3642 / 0.3650	0.451 / 0.4156	0.4115 / 0.3927	0.3467 / 0.3139
48		3COSMUL	---	---	---	---
49	48D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.2393 / 0.2428	0.4693 / 0.4289	0.3642 / 0.3444	**0.3755** / 0.3292
50		cosine distance	0.2521 / 0.2473	0.4704 / 0.4266	0.3694 / 0.3452	0.3919 / 0.3405
51	48D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	0.2899 / 0.4785	0.2892 / 0.4452	0.2895 / 0.4508	0.1842 / 0.2790
52		cosine distance	0.5570 / **0.5988**	0.3932 / 0.4491	0.4675 / **0.5170**	0.3098 / 0.3309
55	24x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.2345 / 0.2426	0.4517 / 0.4127	0.3532 / 0.3356	0.3432 / 0.3044
56		cosine distance	0.1296 / 0.1451	0.2256 / 0.1970	0.1820 / 0.1735	0.1623 / 0.1445
57	24x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	0.4687 / 0.4881	**0.4872** / 0.4513	0.4788 / 0.4680	0.3480 / 0.3136
58		cosine distance	0.2502 / 0.2499	0.2646 / 0.2452	0.2581 / 0.2473	0.1684 / 0.1395
59						
60	**20D Glove embedding results**					
61	20D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	0.1234 / 0.1202	0.2133 / 0.2004	0.1724 / 0.1640	0.1481 / 0.1281
62		3COSMUL	---	---	---	---
63	20D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.1067 / 0.1058	0.2293 / 0.2090	0.1717 / 0.1583	**0.1811** / 0.1579
64		cosine distance	0.1033 / 0.1016	0.2268 / 0.2051	0.1692 / 0.1539	**0.1817** / 0.1554
65	20D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	0.1743 / 0.1840	0.2717 / 0.2793	0.2268 / 0.2321	0.1580 / 0.1475
66		cosine distance	**0.2133** / 0.2018	**0.2950** / 0.2762	**0.2579** / 0.2424	**0.1821** / 0.1611
69	10x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.1009 / 0.1018	0.2111 / 0.1925	0.1611 / 0.1514	0.1593 / 0.1389
70		cosine distance	0.0428 / 0.0394	0.0775 / 0.0690	0.0618 / 0.0556	0.0521 / 0.0446
71	10x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	0.1656 / 0.1629	0.2575 / 0.2313	0.2158 / 0.2003	0.1604 / 0.1452
72		cosine distance	0.0607 / 0.0555	0.1015 / 0.0867	0.0829 / 0.0725	0.0509 / 0.0403
73						
74	**4D Glove embedding results**					
75	4D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	0.0036 / 0.0045	0.0012 / 0.0015	0.0023 / 0.0028	0.0011 / 0.0012
76		3COSMUL	---	---	---	---
77	4D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	0.0090 / 0.0097	0.0044 / 0.0043	0.0064 / 0.0067	0.0056 / 0.0056
78		cosine distance	0.0039 / 0.0041	0.0022 / 0.0027	0.0029 / 0.0033	0.0016 / 0.0018
79	4D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	**0.0135 / 0.0141**	**0.0067 / 0.0064**	**0.0096 / 0.0094**	**0.0061 / 0.0059**
80		cosine distance	0.0054 / 0.0051	0.0027 / 0.0031	0.0036 / 0.0039	0.0021 / 0.0020

Figure 5: Full 180K vocabulary, analogy results.

	A	B	P	Q	R	S	T	U	V
1	**Experiment name**		**Restrict vocab**	**Rare Word (Spearman corr.) using w/w+c**	**WordSim (Spearman corr.) using w/w+c**	**SimLex (Spearman corr.) using w/w+c**	**SimVerb (Spearman corr.) using w/w+c**	**MC (Spearman corr.) using w/w+c**	**RG (Spearman corr.) using w/w+c**
46	**48D Glove embedding results**								
47	48D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	50,000	**0.4299** / 0.4133	**0.6171 / 0.6165**	0.2777 / 0.2756	0.1641 / 0.1583	0.7262 / 0.7438	0.6739 / 0.6931
48		3COSMUL							
49	48D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.4191 / 0.4233	0.6070 / 0.5820	0.2682 / 0.2636	0.1694 / 0.1674	0.7566 / 0.7874	0.6973 / 0.7139
50		cosine distance		0.3896 / 0.3820	0.5760 / 0.5727	0.2590 / 0.2547	0.1568 / 0.1475	0.6871 / 0.6938	0.6494 / 0.6586
51	48D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.3808 / 0.3951	0.4940 / 0.5116	0.2449 / 0.2507	0.1607 / **0.1703**	0.7334 / **0.8352**	0.6982 / 0.7366
52		cosine distance		0.3921 / 0.3712	0.5719 / 0.5714	0.2721 / 0.2672	0.1574 / 0.1480	0.7031 / 0.7189	0.6661 / 0.6827
53	24x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.4235 / 0.4231	0.6044 / 0.5973	**0.2790** / 0.2764	0.1636 / 0.1647	0.7834 / 0.7908	0.7294 / **0.7451**
54		cosine distance		0.3425 / 0.3339	0.5169 / 0.5208	0.2500 / 0.2297	0.1255 / 0.1232	0.7002 / 0.7248	0.6049 / 0.6110
55	24x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.4121 / 0.4058	0.5759 / 0.5677	0.2703 / 0.2695	0.1601 / 0.1635	0.7911 / 0.8076	0.7302 / 0.7436
56		cosine distance		0.3702 / 0.3627	0.5470 / 0.5372	0.2371 / 0.2392	0.1351 / 0.1347	0.7199 / 0.7026	0.6475 / 0.6046
57									
58	**20D Glove embedding results**								
59	20D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	50,000	0.3695 / 0.3575	0.5198 / **0.5223**	**0.2426** / 0.2363	0.1271 / 0.1233	0.6683 / 0.6883	0.5960 / 0.6242
60		3COSMUL							
61	20D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.3683 / 0.3608	0.4913 / 0.4784	0.2255 / 0.2246	0.1317 / 0.1339	0.6627 / 0.6662	0.6384 / 0.6631
62		cosine distance		0.3356 / 0.3270	0.4677 / 0.4638	0.2175 / 0.2157	0.1218 / 0.1207	0.5896 / 0.6125	0.5656 / 0.5821
63	20D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.3355 / 0.3431	0.4125 / 0.4107	0.2100 / 0.2179	0.1240 / 0.1382	0.6603 / **0.7186**	0.6556 / **0.6997**
64		cosine distance		0.3510 / 0.3345	0.4861 / 0.4875	0.2324 / 0.2314	0.1187 / 0.1172	0.6445 / 0.6526	0.5993 / 0.6240
65	10x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.3749 / 0.3652	0.4893 / 0.4827	0.2321 / 0.2325	0.1254 / 0.1300	0.6775 / 0.6721	0.6367 / 0.6538
66		cosine distance		0.3059 / 0.2976	0.4268 / 0.4206	0.1709 / 0.1780	0.0747 / 0.0749	0.5928 / 0.5698	0.5358 / 0.5635
67	10x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	**0.3771** / 0.3637	0.4748 / 0.4782	**0.2438** / 0.2338	**0.1396 / 0.1384**	0.6502 / 0.7132	0.6461 / 0.6779
68		cosine distance		0.3249 / 0.3196	0.4352 / 0.4440	0.1935 / 0.1743	0.0860 / 0.0812	0.5211 / 0.5718	0.4853 / 0.5170
69									
70	**4D Glove embedding results**								
71	4D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	50,000	0.1744 / 0.1747	0.2113 / 0.2130	0.147 / 0.1473	0.0582 / 0.0582	0.3227 / 0.3161	0.3973 / 0.3953
72		3COSMUL							
73	4D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.2183 / 0.2185	0.2799 / 0.2799	0.1530 / 0.1531	0.0745 / 0.0746	0.4104 / 0.4124	0.4548 / 0.4523
74		cosine distance		0.1917 / 0.1915	0.1765 / 0.1762	0.1441 / 0.1442	0.0673 / 0.0674	0.2927 / 0.3020	0.4681 / 0.4650
75	4D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.2265 / 0.2269	0.2357 / 0.2372	0.1273 / 0.1274	0.0605 / 0.0608	0.2784 / 0.2698	0.3495 / 0.3520
76		cosine distance		0.1743 / 0.1740	0.1748 / 0.1770	0.1005 / 0.1007	0.0451 / 0.0452	0.1826 / 0.1860	0.2979 / 0.3016

Figure 6: Restricted 50K (most frequent) vocabulary, similarity results.

	A	B	P	Q	R	S	T
1	**Experiment name**		**Restrict vocab**	**Semantic Google analogy accuracy using w/w+c**	**Syntactic Google analogy accuracy using w/w+c**	**Total Google analogy accuracy using w/w+c**	**MSR analogy accuracy using w/w+c**
46	**48D Glove embedding results**						
47	48D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	50,000	0.3212 / 0.3299	0.4727 / 0.4303	0.4039 / 0.3847	0.355 / 0.3156
48		3COSMUL		---	---	---	---
49	48D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.2127 / 0.2178	0.4683 / 0.4302	0.3521 / 0.3338	0.3581 / 0.3078
50		cosine distance		0.2200 / 0.2236	0.4690 / 0.4261	0.3551 / 0.3342	**0.3708** / 0.3134
51	48D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.2903 / 0.3799	0.2666 / 0.4129	0.2774 / 0.3923	0.1787 / 0.2652
52		cosine distance		0.4075 / **0.4380**	0.4102 / 0.4394	0.4090 / **0.4388**	0.3041 / 0.3141
53	24x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.2309 / 0.2308	0.4512 / 0.4096	0.3513 / 0.3285	0.3295 / 0.2976
54		cosine distance		0.1324 / 0.1341	0.2483 / 0.2153	0.1957 / 0.1784	0.1860 / 0.1554
55	24x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.3668 / 0.3800	**0.4749** / 0.4374	0.4259 / 0.4113	0.3451 / 0.3008
56		cosine distance		0.2055 / 0.2102	0.2707 / 0.2496	0.2411 / 0.2317	0.1820 / 0.1624
57							
58	**20D Glove embedding results**						
59	20D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	50,000	0.1223 / 0.1164	0.2472 / 0.2120	0.1905 / 0.1686	0.1550 / 0.1289
60		3COSMUL		---	---	---	---
61	20D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.0925 / 0.0903	0.2445 / 0.2124	0.1743 / 0.1548	0.1616 / 0.1338
62		cosine distance		0.0917 / 0.0890	0.2448 / 0.2116	0.1733 / 0.1533	0.1644 / 0.1324
63	20D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.1615 / 0.1661	0.2725 / 0.2603	0.2149 / 0.2112	0.1624 / 0.1454
64		cosine distance		0.1757 / **0.1777**	**0.2970** / 0.2613	**0.2408** / 0.2220	**0.1804** / 0.1554
65	10x2D, MIX-Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.0958 / 0.0943	0.2199 / 0.1926	0.1636 / 0.1480	0.1554 / 0.1279
66		cosine distance		0.0400 / 0.0387	0.0860 / 0.0709	0.0651 / 0.0563	0.0490 / 0.0434
67	10x2D, MIX-Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.1443 / 0.1475	0.2582 / 0.2281	0.2065 / 0.1915	0.1598 / 0.1444
68		cosine distance		0.0602 / 0.0581	0.1030 / 0.0901	0.0836 / 0.0756	0.0473 / 0.0437
69							
70	**4D Glove embedding results**						
71	4D, Vanilla GloVe, f(dist)=dot product, 50ep, sim=cosine distance	3COSADD	50,000	0.005 / 0.0053	0.0011 / 0.0012	0.0029 / 0.0031	0.0011 / 0.0011
72		3COSMUL		---	---	---	---
73	4D, Poincare GloVe, f(dist)=cosh-dist-sq, 50ep, radagrad	Poincare distance	50,000	0.0058 / 0.0058	0.0042 / 0.0044	0.0047 / 0.0049	0.0043 / 0.0044
74		cosine distance		0.0048 / 0.0039	0.0027 / 0.0030	0.0036 / 0.0034	0.0010 / 0.0009
75	4D, Poincare GloVe, f(dist)=dist-sq, 50ep, radagrad	Poincare distance	50,000	0.0133 / 0.0133	0.0076 / 0.0078	0.0102 / 0.0103	0.0061 / 0.0065
76		cosine distance		0.0069 / 0.0068	0.0035 / 0.0042	0.0047 / 0.0054	0.0026 / 0.0027

Figure 7: Restricted 50K (most frequent) vocabulary, analogy results.

B δ-HYPERBOLICITY

Let us start by defining the *δ-hyperbolicity*, introduced by Gromov (1987). The *hyperbolicity* $\delta(x, y, z, t)$ of a 4-tuple (x, y, z, t) is defined as half the difference between the biggest two of the following sums: $d(x, y) + d(z, t)$, $d(x, z) + d(y, t)$, $d(x, t) + d(y, z)$. The δ-hyperbolicity of a metric space is defined as the supremum of these numbers over all 4-tuples. Following Albert et al. (2014), we will denote this number by δ_{worst}, and by δ_{avg} the average of these over all 4-tuples, when the space is a finite set. An equivalent and more intuitive definition holds for geodesic spaces, *i.e.* when we can define triangles: its δ-hyperbolicity (δ_{worst}) is the smallest $\delta > 0$ such that for any triangle Δxyz, there exists a point at distance at most δ from each side of the triangle. Chen et al. (2013) and Borassi et al. (2015) analyzed δ_{worst} and δ_{avg} for specific graphs, respectively. A low hyperbolicity of a graph indicates that it has an undelrying hyperbolic geometry, *i.e.* that it is approximately tree-like, or at least that there exists a taxonomy of nodes. Conversely, a high hyperbolicity of a graph suggests that it possesses long cycles, or could not be embedded in a low dimensional hyperbolic space without distorsion. For instance, the Euclidean space $\mathbb{R}^n$ is not δ-hyperbolic for any $\delta > 0$, and is hence described as ∞-hyperbolic, while the Poincaré disk $\mathbb{D}^2$ is known to have a δ-hyperbolicity of $\log(1 + \sqrt{2}) \simeq 0.88$. On the other-hand, a product $\mathbb{D}^2 \times \mathbb{D}^2$ is ∞-hyperbolic, because a $2D$ plane $\mathbb{R}^2$ could be isometrically embedded in it using for instance the first coordinates of each $\mathbb{D}^2$. However, if $\mathbb{D}^2$ would constitute a good choice to embed some given symbolic data, then most likely $\mathbb{D}^2 \times \mathbb{D}^2$ would as well. This stems from the fact that δ-hyperbolicity (δ_{worst}) is a worst case measure which does not reflect what one could call the "hyperbolic capacity" of the space. Furthermore, note that computing δ_{worst} requires $\mathcal{O}(n^4)$ for a graph of size n, while δ_{avg} can be approximated via sampling. Finally, δ_{avg} is robust to adding/removing a node from the graph, while δ_{worst} is not. For all these reasons, we choose δ_{avg} as a measure of hyperbolicity.

C CLOSED-FORM FORMULAS OF MÖBIUS OPERATIONS

We show closed form expressions for the most common operations in the Poincaré ball, but we refer the reader to (Ungar, 2008; Ganea et al., 2018b) for more details.

Möbius addition. The *Möbius addition* of x and y in $\mathbb{D}^n$ is defined as

$$x \oplus y := \frac{(1 + 2\langle x, y\rangle + \|y\|^2)x + (1 - \|x\|^2)y}{1 + 2\langle x, y\rangle + \|x\|^2\|y\|^2}. \tag{8}$$

We define $x \ominus y := x \oplus_c (-y)$.

Möbius scalar multiplication. The *Möbius scalar multiplication* of $x \in \mathbb{D}^n \setminus \{\mathbf{0}\}$ by $r \in \mathbb{R}$ is defined as

$$r \otimes x := \tanh(r \tanh^{-1}(\|x\|)) \frac{x}{\|x\|}, \tag{9}$$

and $r \otimes \mathbf{0} := \mathbf{0}$.

Exponential and logarithmic maps. For any point $x \in \mathbb{D}^n$, the exponential map $\exp_x : T_x\mathbb{D}^n \to \mathbb{D}^n$ and the logarithmic map $\log_x : \mathbb{D}^n \to T_x\mathbb{D}^n$ are given for $v \neq \mathbf{0}$ and $y \neq x$ by:

$$\exp_x(v) = x \oplus \left(\tanh\left(\frac{\lambda_x\|v\|}{2}\right)\frac{v}{\|v\|}\right), \ \log_x(y) = \frac{2}{\lambda_x}\tanh^{-1}(\|-x \oplus y\|)\frac{-x \oplus y}{\|-x \oplus y\|}. \tag{10}$$

Gyro operator and parallel transport. Parallel transport is given for $x, y \in \mathbb{D}^n, v \in T_x\mathbb{D}$ by the formula $P_{x\to y}(v) = \frac{\lambda_x}{\lambda_y} \cdot \text{gyr}[y, -x]v$. Gyr[4] is the gyroautomorphism on $\mathbb{D}^n$ with closed form expression shown in Eq. 1.27 of (Ungar, 2008):

$$\text{gyr}[u, v]w = \ominus(u \oplus v) \oplus \{u \oplus (v \oplus w)\} = w + 2\frac{Au + Bv}{D} \tag{11}$$

[4] https://en.wikipedia.org/wiki/Gyrovector_space

where the quantities A,B,D have closed form expression and, thus, are easy to implement.

$$A = -\langle u, w\rangle \|v\|^2 + \langle v, w\rangle + 2\langle u, v\rangle \cdot \langle v, w\rangle \tag{12}$$

$$B = -\langle v, w\rangle \|u\|^2 - \langle u, w\rangle \tag{13}$$

$$D = 1 + 2\langle u, v\rangle + \|u\|^2 \|v\|^2 \tag{14}$$