# Joint Label/Example Hyperbolic Representation for Extreme Classification

Thomas Gerald, Nicolas Baskiotis

# Joint Label/Example Hyperbolic Representation for Extreme Classification

Thomas Gerald and Nicolas Baskiotis

Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

## Abstract

Extreme multi-label classification task deals with nowadays problems involving up to millions of labels. The extremely large number of labels requires efficient methods not only in terms of prediction performances but also in terms of time processing and memory management. Most recent works focus on representation learning approaches: embeddings allow to infer a semantic structured space with interesting generalization properties; the label prediction is often performed by using approximated nearest neighbors search. Recent works in Language Modeling and especially in Question Answering have shown increasing performances using the hyperbolic space to learn the representation. The Poincaré ball model is indeed more relevant to represent ontology than the euclidean space. In this work, we propose to explore representation learning in hyperbolic space in the context of extreme classification. The proposed model performs a joint embedding of examples and labels. Most of the embedding methods for extreme classification only learn example embedding. Thus in order to structure the space, they require to precompute examples neighborhood. In our approach, learning a joint embedding allows structuring the space without any expensive preprocessing. Experiments conducted on real-world datasets show the performances of our model compared to state of the art.

## 1 Introduction

Large-Scale Multi-Label classification also known as *Extreme Classification* has become a real challenge of interest this last decade. Its objective is to develop classification algorithms able to identify multiple relevant labels among a very large label set for an example given as input. Large-scale classification is different due to the large label and features dimension, especially regarding the number of labels to discriminate. Assigning automatically topics/labels for a large amount of data collected by social network or encyclopedia is a main subject of interest. Even in the industry such as e-commerce platforms, finding relevant object associated with a query is an important task and requires often automatic tagging. Recently the domain has grown significantly and most leading methods are based on representation learning approaches. Finding relevant representation is, therefore, a key task to tackle *extreme classification*. In extreme classification leading methods are often based on tree involving to discover latent hierarchy. Representation models do not include this hierarchical information, however recent works on embedding hierarchy using hyperbolic representation space have proven their efficiency to capture the latent hierarchical information.

In this work, we propose to investigate representation learning for *Extreme Multi-Label Classification* by using instead of the usual Euclidean metric space the Poincaré ball model. To structure the embedding space we learn a joint label/example representation. We study the structure of the learned space and show experimentally that it is a relevant alternative to the euclidean space for multi-label classification. We present experiments on several large scale datasets and compare our approach to state of the art models. We show that for certain types of corpus our approach is relevant, especially when the number of dimensions of the embedding space is low.

Classification of documents, images or items has raised interest during these previous years due to the spread of available data, the diversity of new methods and the growing of computational power. Document classification has benefited from many improvements reaching impressive performances. However, classification of documents can be inefficient in computation time, prediction performances and memory management when the number of labels to discriminate be-

come too large. To address this issue, the Extreme Multi-label Classification domain is an active research field to deal with the extremely large number of classes (from $10^4$ up to $10^7$).

Algorithms for *Extreme multi-label classification* follow mainly three paradigms: 1) One Versus Some/All paradigm (OVA); 2) Hierarchical paradigm using classifiers organized by a graph structure; 3) Representation learning paradigm.

Historical baselines dealing with multi-label setting are mainly based on the *One versus All* paradigm: for each label present in the dataset a classifier is learned to distinguish it from the others; the prediction step infers the relevant labels using a voting scheme mixing the outputs of all the classifiers. The approach is extremely costly in terms of computation time and in memory storage, needing an extremely large number of classifiers growing linearly with respect to the number of labels. Another major drawback of this paradigm is its incapacity to exploit label correlation [ZLLG18]. To reduce the computation cost, methods using sparsity have been proposed such as `PDSparse` [YHZ$^+$, YHD$^+$17] using weight regularization to ensure the sparsity of each classifier [ZH05]. With the growth of parallel computational power, distributed `OVA` methods `DiSmec` success to learn classifiers efficiently using 1000 CPU cores, and manage the memory/time tradeoff by integrating pruning methods.

Hierarchical approaches exploit the divide and conquer paradigm to reduce drastically the number of classifiers used to infer labels. The classifiers are embedded in a tree structure, where each split corresponds to a split of the label set. The classifier attached to a given node is fitted in order to recognize at which child label subset belongs an example. Each leaf of the tree corresponds ideally to a unique label. The classification process begins at the root note and follows a path through the tree until a leaf is reached. The complexity of the inference is thus logarithmic with respect to the number of classes. Although finding an optimal tree structure and label organization is a complex problem, the recent approaches `FastXML` and `PfastreXML`[PV14, JPV16] reach the state of the art performances. *FastXML* recursively partitioning nodes maximizing the *Normalized Discounted Cumulative Gain* (`nDCG`) ranking loss; *PFastreXML* uses for the partitioning a new loss function, the *Propensity scored Loss*. This loss is also used to evaluate performances taking into account the frequencies of labels in corpora. Still using the divide and conquer paradigm, the recent method `PARABEL`[PKH$^+$18] merges OVA and hierarchical approach proposing overlapping label partitioning at each node.

## 1.1 Learning embedding for extreme classification

*Label dimension reduction* approaches reduce the complexity of the inference step by taking into account the label correlations.

A first approach consists to factorize the label matrix, the label matrix being the matrix containing for each example a label vector with value 1 if labels are annotated with. Remarking that labels can be correlated led to find a way to factorize the matrix. Thus, the labels matrix can be embedded in a sub-dimensional space. This hypothesis is named the low-rank assumption, it has been applied in [HKLZ09, TL12] by factorizing the matrix using eigenvectors with the largest eigenvalues. Similarly, methods have been proposed to reduce the feature space dimension [CL12]. However, in the case of extreme classification, the low-rank assumption is very naive: in case of large diversity, labels are represented only a few times in corpora [Tag17]. Despite the dimensionality reduction, these methods are still time costly especially due to the decoding process within a time complexity still linear in the number of labels.

In order to deal with a large number of labels during the prediction process, several methods have proposed to use nearest neighbors search procedure. The number of label candidates is thus drastically reduced as only the labels of nearest examples are taken into account. For instance, the *SLEEC* approach [BJK$^+$15] operates first partitioning of the input space and then learns mapping features to embeddings in each partition according to a label/example similarity. The inference is finally done by retrieving the labels of the nearest neighbors of the input example. This method has impressive performances on many corpora. The more recent method *AnnexML*[Tag17] proposes several improvements, particularly, it introduces a new partitioning scheme taking into account the label similarity, a new loss function to define the similarity between examples and an ensemble learning scheme to improve performances. The inference time is also drastically lowered thanks to an approximate nearest neighbor search.

## 1.2 Hyperbolic representation space

Using hyperbolic geometry to embed hierarchical information has recently raised the interest of the machine learning community.

Particularly the Poincaré ball model having good properties such as differentiable distance or hierarchical compliant geometry. The point at the origin of the ball is thus a natural candidate to represent the top hierarchy and the edge of the ball the leaves. Several recent works use this space: [NK17] have proposed an embedding of word vectors with most general concepts closer to the center of the ball;

*Ganea et al*[SDSGR18] have shown the efficiency of the representation space in extremely low dimension, producing state of the art results for *WordNet* embeddings using only two-dimensional latent space, but with complete hierarchy known. Moreover, it has been empirically proven to be efficient in not Tree like corpora. If previous methods are directly learning to embed without parametric functions, learning a mapping function from a features space has also been studied.

The algorithm *HyperQA* [TLH17] have reached state of the art on Question-Answering domain by using neural-networks to produce Poincaré Ball embedding.

They also empirically show that using Poincaré ball model for information retrieval through parametric function can lead to state of the art performances and confirm the low dimension efficiency. More recently the works *Hyperbolic Neural Networks*[GBH18] provided a set of neural layers for deep learning approaches using hyperbolic structure.

If learning representation for large scale multi-label classification remains a relevant approach, capture the hierarchical structure of data is not covered by today *Extreme classification* representation methods. Indeed, it is difficult to observe and embed hierarchy in Euclidean space while the majority of large scale corpora have a latent taxonomy. Thus using Hyperbolic representation space could lead to disambiguate example/labels hierarchy, where Euclidean can not. Moreover, due to the robustness of those model using low dimension, nearest neighbors search prediction could be more stable using a lower dimension. Thus leading to obtaining the same performances faster because of lower dimensionality.

# 2 Proposed approach

In this work we propose to study a model mapping the *feature* space and the *labels* into a joint hyperbolic representation space. Thus we design two functions, one for the feature embedding and the other one for the label embedding. In order to learn these representations jointly, we design a loss that tends to set closer the embedding of an example and the embedding of a label when the example belongs to the given label. Once representation learned examples representation within similar labels tends to be close, thus we perform prediction by annotating labels of an example according to its neighbors in the representation space. The section is organized as follows: first we introduce the notation being used; next we present a quick introduction to the *Poincaré Ball* model and give insight about the geometry; the model architecture and the associated learning procedure are next presented; finally, we present the inference step.

## 2.1 Notations

Let $\mathbf{X} \in \mathbb{R}^{N \times m}$ be the data matrix containing $N$ examples lying in a $m$ dimensional feature space; $x_i \in \mathbb{R}^{\mathbb{N}}$ the $i$-th example; $\mathbf{Y} \in \{0,1\}^{N \times L}$ the associated label matrix with $L$ labels, such that $y_{i,k} = 1$ if the example $i$ is annotate with label $k$ (otherwise $y_{i,k} = 0$). We will note $y_i$ the $i$-th line of the label matrix, corresponding to the label vector of the example $x_i$. We aim to embed both examples $x_i$ and labels $k \in \{1, 2, 3, \ldots, L\}$ in the Poincaré ball model.

## 2.2 Poincaré Ball Model

The Poincaré ball model is a model of hyperbolic geometry lying in an n-dimensional sphere. It can be intuitively seen as the projection of a hyperbolic function of dimension $n+1$ to a ball of dimension $n$. The original model maps points of the hyperbolic function in the open unit ball. A particularity of this space is that the norm of a vector tends to infinity when the vector tends to the boundary of the ball. Thus a point near the ball center tends to be closer to all points than points near the boundary. This is explained by the formulation of the metric in the space. This property makes the Poincaré ball model a relevant choice for embedding hierarchical data by mapping the nodes near the root closer to the origin and leaves near the boundary, as represented in the figure 2.2. Each segment in the figure has the same length in the hyperbolic space.

Formally, let be $\mathcal{B}^n = \{x \in \mathbb{R}^n | \, \|x\| < 1\}$ the n-dimensional open unit ball. We refer now as $\mathbb{H}_n$ the Poincaré ball model of dimension $n$ corresponding to the metric space on $\mathcal{B}^n$ equipped with the metric $g_x$, i.e $(\mathcal{B}^n, g_x)$. The $g_x$ metric is defined as (with $g^E$ the euclidean metric tensor):

$$g_x = \left( \frac{2}{1 - \|x\|_2^2} \right)^2 g^E$$

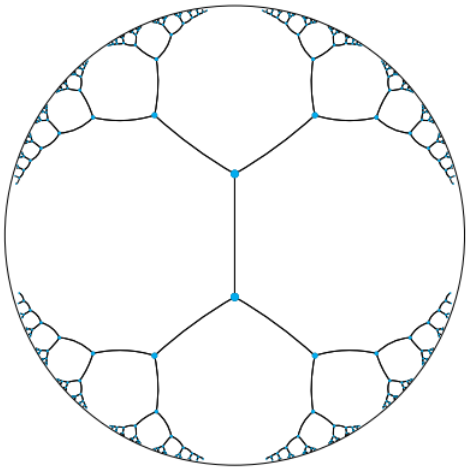

Figure 1: Embedding of a tree in $\mathcal{B}^2$ (This figure come from the paper [NK17])

The definition of the hyperbolic metric space depicts that a point will have a hyperbolic norm depending on the inverse of the square of the Euclidean norm: embedding close to the boundary of the ball will have a norm tending to infinity. The hyperbolic distance in the *Poincaré ball* model $d$ is defined for two vectors $x_1, x_2 \in \mathbb{H}_n$ as following :

$$d(x_1, x_2) = arcosh\left(1 + 2\frac{\|x_1 - x_2\|_2^2}{(1 - \|x_1\|_2^2)(1 - \|y_1\|_2^2)}\right)$$

The distance is differentiable, thus usual back-propagation algorithms can be used while taking into account the curvature of the space into the optimization procedure.

## 2.3 Embedding into the Poincaré ball model

Our objective is to embed in the same space labels and examples in hyperbolic space in such a way that examples with similar annotation being close to each other. As there is no mapping function to learn for the labels, we represent their embeddings by a matrix $W^l \in \mathbb{R}^{L \times n}$, with $L$ the number of labels and $n$ the dimension of the Poincaré ball, i.e. the size of the latent representation. The $i$-th line of the matrix $W^l$, $l_i \in \mathbb{R}^n$, correspond to embedding of the $i$-th label . In order to learn the mapping between the feature space and the hyperbolic space, we use a simple linear projection layer $f_\theta : \mathbb{R}^M \to \mathbb{R}^n$ with $\theta \in \mathbb{R}^{M \times n}$. Let $r_j$ the projection of the example $j$ obtained by the function $f_\theta$:

$$r_j = f_\theta(x_j) = \theta' \cdot x_j$$

More complex mapping functions can be used (for instance neural networks with multiple layers) but experiments showed that a linear projection is sufficient to reach the best performances.

In order to constrain the projection of the labels and the examples to the n-dimensional unit ball, we define the following projection:

$$p(x) = \begin{cases} \frac{x}{\|x\|+\varepsilon} & \text{if } \|x\| \geq 1 \\ x & \text{otherwise} \end{cases}$$

Thus the label for the $i$-th label is given by $p(l_i)$ and embedding of an example $x_j$ by $p(f_\theta(x_j)) = p(r_j)$.

## 2.4 Loss function

Most of recent contributions in representation learning for extreme classification are trying to obtain close example embeddings for similar examples according to their label vectors. For instance, *AnnexML* [Tag17] proposes a similarity between examples based on the cosine similarity of the vector labels:

$$cos(y_i, y_j) = \frac{y_i \cdot y_j}{\|y_i\|_2 \|y_j\|_2}$$

with $y_i$ the label vector of $x_i$. Today approaches to structure the embedding space is to first for each example define a set of neighbors based on a label similarity and then design a cost function setting closer examples within the same neighborhood.

The following loss is used in [Tag17] for a given example $x_j$ and a example $x_v$ in the neighborhood $V_j$ of $x_j$ (i.e. the $k$-nearest neighbors of the example $x_j$ with respect to the cosine similarity between the label vectors), and $r_i$ the euclidean embedding associate to the example $x_i$:

$$L(x_j, x_v) = -log(\frac{e^{\alpha cos(r_j, r_v)}}{\left[\sum\limits_{k \notin V_j} e^{\alpha cos(r_j, r_k)}\right] + e^{\alpha cos(r_j, r_v)}})$$

With $k \notin V_j$ denote examples $x_k$ who do not belong to the $x_j$ neighborhood and $\alpha$ controlling the smoothness of the softmax loss.

This loss model the negative log-probability of having embedding of $x_v$ belonging to the neighborhood of $x_j$ embedding.

This method requires before the learning step to process the dataset using nearest neighbors search in order to get for each example their neighborhood.

Contrary to the state of the art methods, we propose to learn the example embeddings based on the label embedding, by setting closer example embeddings using label representation. This is made possible because both representations lying in the same space. Thus we do not need to pre-compute any label correlations or pre-computing neighborhood for the examples. The loss proposed in this work is similar to the previous one, but instead of processing distance between the example representations, we compute the distance of the representation of an example to the representation of one of its labels. Let $V(j) = \{k|y_{jk} = 1\}$ the labels of the example $x_j$; the proposed loss for an example $x_j$ and a label $i$ such that $i \in V(j)$ is :

$$L(x_j, l_i) = -log(\frac{e^{-\alpha d(r_j, l_i)}}{\sum\limits_{k \notin V(j)} e^{-\alpha d(r_j, p(l_k))} + e^{-\alpha d(r_j, p(l_i))}})$$

with $r_j = p(f_\theta(x_j))$ the projection of $x_j$ and $l_i$ the representation of the label $i$. Similarly, this can be interpreted as the negative log-probability of having the label $i$ rather than another for the example $x_j$. When the convergence is achieved the label space has embedded closer examples with shared labels, and the representation of the examples should be close to the representation of their labels. Moreover, we expect that examples having the most label in common will be rather close to each other in terms of hyperbolic distance. In practice, for an example $x_j$, we use negative sampling by randomly sampling labels rather than considering all labels which are not in $V(j)$. Because of the large number of labels, there is only a low probability to sample a label belonging to $V(j)$. Due to the structure of the hyperbolic representation space, rare labels should be close to the boundary of the ball. At the opposite, top labels appearing often in the dataset should closer to the origin.

### 2.5 Optimization algorithm

All the presented function are differentiable, however, we can not learn that representation using regular gradient descent algorithm due to the hyperbolic embedding: the gradient depends on the metric $g_x$. In order to take into account the slope of the space curvature, the paper [NK17] has proposed a gradient descent algorithm named projected *RSGD* which scale the gradient according to the metric of the space. To update the label embeddings the gradient obtained by backpropagation is scaled by the inverse of the metric $g_x^{-1}$. Considering at a step $t$ of the algorithm the representation vector $l_i^t$ of the $i$-th label, the update of the representation at $t+1$ is given by :

$$l_i^{t+1} = l_i^t - \eta \left( \frac{1 - \|l_i^t\|_2^2}{2} \right)^2 \nabla_{p(l_i)} L(x_i, l_i)$$

with $\epsilon$ the learning rate. In a similar way, the update of the parameter $\theta^t$ of the mapping function of the examples is given by the following rule :

$$\theta^{t+1} = \theta^t - \eta \nabla_{\theta^t}(r_j) \left( \frac{1 - \|r_j\|_2^2}{2} \right)^2 \nabla_{r_j} L(x_j, l_i)$$

Experimentally we obtained better results using the *Adam* optimization algorithm applying the previous inverse norm factor on the value to update, however, due to exponentials decays in *Adam* we can not consider this method as formal.

### 2.6 Inference step

To annotate a new example $x$ we embed the example to the representation space with $p(f_\theta(x))$. Then an *approximate K Nearest Neighbor Search* according to the distance $d$ is performed to retrieve closest training examples. The number of occurrences of each label associate to embedding within the neighborhood of $x$ representation is aggregated and ranked accordingly. The most relevant labels correspond to the most occurring labels in the neighborhood, with the highest apparition frequency.

To perform the approximate neighbor search, we experimentally observe that random selection of centroids among the representations of the training examples to construct the clusters does not decrease performances in comparison to exact neighbors search.

The approximate search allows to largely decrease the inference time. In the remainder of this paper, we will use $\frac{N}{3000}$ random clusters by default.

## 3 Experiments

### 3.1 Experimental settings

Table 1 presents the selected datasets to evaluate the proposed method. All of them are real-world corpora based on Wikipedia annotations (*Wiki10-31K, WikiLSHTC-325K*), web page annotations (*Delicious-200K*) or web market items/query annotations (*AmazonCat-13K, Amazon-670K*). The datasets have major differences: the Wikipedia and the web page description dataset have a hierarchical label organization while the two amazon datasets have a more flat label organization; the number of labels

by example and the number of examples for a given label also differ from a factor larger than 10 between the datasets. Moreover, corpus as *wiki10* contains very few examples compared to the number of labels, thus the number of labels will be rarely considered during learning. At the contrary in the case of *AmazonCat* where the number of examples per label is large, most of the labels will be updated many times.

In the following experiments, when not specified the selected dimension of the representation space is generally fixed to 50 (like in most multi-label classification works [BJK+15, Tag17]). A validation set is used to control the overfitting, build by sampling 5% of the training set. The learning process is stopped when precision does not increase during 25 epochs. The label embedding matrix is initialized according to a normal distribution $\mathcal{N}(0, 1e-3)$ and the mapping function $f\_theta$ with the distribution $\mathcal{N}(0, 1e-4)$.

Concerning the optimization, the *adam* optimizer [KB14] is used with hyperbolic Poincaré retraction (see 2.5), which shown experimentally better results than the regular stochastic descent. The main evaluation metric considered is the *precision at k* (p@k): given $\hat{y}_i$ the ordered label prediction inferred for an example $x_i$, with $\hat{y}_i^j \in \{1, 2, \ldots, L\}$ the $j$-th predicted label, the precision is computed as follows:

$$p@k = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{k} \sum_{j=1}^{k} y_{i,\hat{y}_i^k}$$

Table 2 presents a naive baseline inspired by [Tag17] showing the difference between the datasets regarding the label distribution. The baseline named *MostCommon*, predicts the same ordered label vector for all the examples, sorted from the most frequent label in the training dataset to the less frequent one (e.g in *wiki10*, the five most common labels are *wikipedia, wiki, reference, history, research*).

## 3.2 Evaluation

Table 3 shows the results of the proposed approach (*Hyp*) compared to the state of the art results. Outside our results, the reported results are from the *Extreme classification repository*[1] except for the *AnnexML* algorithm for which the results come from [Tag17]. Following their experimental settings, the predictions are made by an ensemble learning model, averaging the vote of 15 different models. The performances differ widely with respect to the nature of the dataset: on *Wiki10* and *delicious-200K*, our method is very close to the best results; on the other datasets, there is a gap between the performances of our approach and the others. The main difference between the two sets of corpora is the hierarchical latent organization of the labels: the performances are better with our approach when such hierarchical organization is deep like for *Wiki10* or *Delicious*. At the contrary, results with more flat hierarchies like *Amazon* did not succeed to reach competitive performances.

**Effect of the embedding dimension** Poincaré ball model is known for the ability to efficiently embed hierarchical element in a low dimensional space. We design an experiment to verify this hypothesis by training our model using different embedding sizes and by comparing it to the *AnnexML* algorithm without prepartitioning (corresponding in fact to an euclidean embedding). Table 4 shows the results for *wiki10*. For low dimensions (5 or 10), our approach slightly outperforms the euclidean embedding especially in the case of a single learner. With the growth of the number of dimensions, euclidean embedding tends to outperform our approach.

The dimension of the embedding space is crucial in the case of extreme classification, as it is a key factor for the inference time: the k-nearest neighbors algorithm used for the prediction is linear with respect to the embedding size, thus the complexity of the inference step can be drastically improved if a very low embedding size is sufficient for good prediction performances.

**Ensemble Learning** Ensemble learning in large scale multi-label classification is a key process to increase performances. As in the case of the state of the art algorithms, we found that using several learners (models) and aggregating their votes improve the performances. The results are reported in Table 5. Our approach seems to benefit from multiple models, but quickly reach a plateau around 6 learners. These experiments show that the presented approach is quite robust and the learned representations relatively stable compare to *annexML* and *SLEEC* model.

**Analysis of the representation space** In this section, we report qualitative results allowing to understand the properties of the inferred representation space. A first observation concerns the learned representation of the labels: as expected, the norm of the embedding representation of a label is linked to its frequency in the dataset: we observe that most common labels have small norm when the less common ones

[1] http://manikvarma.org/downloads/XC/XMLRepository.html

| Dataset | Avg #examples per label | Avg #labels per example | #labels | #train |
|---|---|---|---|---|
| AmazonCat-13K | 448.57 | 5.04 | 306,782 | 1,186,239 |
| Wiki10-31K | 8.52 | 18.64 | 30,938 | 14,146 |
| Delicious-200K | 72.29 | 75.54 | 205,443 | 196,606 |
| WikiLSHTC-325K | 17.46 | 3.19 | 325,056 | 1,778,351 |
| Amazon-670K | 4 | 5.45 | 670,091 | 490,449 |

Table 1: Main statistics of the dataset: the average number of examples per label, the average number of labels per example, the number of labels and the number of training examples.

| Dataset | p@1/p@3/p@5 |
|---|---|
| AmazonCat-13K | 30.0%/18.8%/14.9% |
| Wiki10-31K | 80.8%/50.5%/36.8% |
| Delicious-200K | 38.7%/36.8%/35.5% |
| WikiLSHTC-325K | 15.9%/6.03%/3.80% |
| Amazon-670K | $2.8e^{-3}$%/$2.7e^{-3}$%/$2.3e^{-3}$% |

Table 2: Most-common baseline results, precision at 1/3/5 when the predicted labels are the most frequent labels of the training set.

have the largest norm. For instance, in the case of *wiki10*, Table 6 shows the norm and the frequency of some labels. This experiment indicates that our model is able to capture the latent hierarchical information among the labels. To confirm this fact, we learned a model in a simple two dimensions space. Figure 3.2 shows the label embeddings learned by this model. The color of the vector is depending on the frequency of the embedded label. The visualization shows that rare labels are very close to the boundary of the ball.

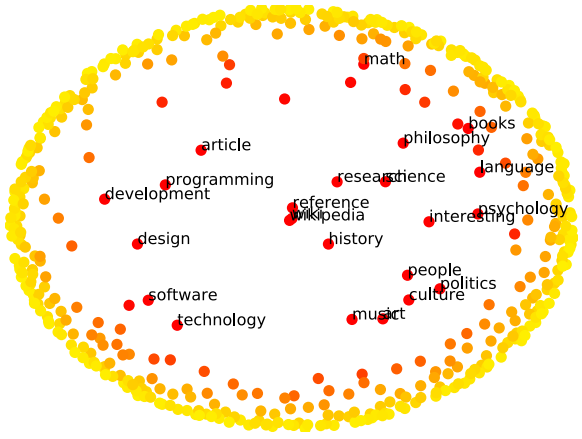


Figure 2: Label embeddings for *wiki10* in a two dimensions representation space. The color tends to red for more frequent labels and to yellow for rare labels.

# 4 Conclusion and Future Works

In this work we investigated the *Extreme Multi-Label classification* task using a hyperbolic representation learning paradigm. We propose an algorithm to learn a joint label/example representation. We investigate the strength of the hyperbolic space to represent a large amount of data and its capacity to represent and capture latent hierarchical information. Our procedure does not involve a preprocessing step when state of the art algorithms use a heavy preprocessing clustering procedure and/or pre-computation of example similarities. The conducted experiments have shown that for datasets with deep hierarchical label organization, our model reaches the state of the art algorithms and can outperform them in very low dimension. This is an important fact as the number of dimensions is a key factor for the inference time. However, on other datasets with a flat hierarchical structure, there is a real gap of performances. The first improvement under investigation concerns the loss function. The experiments show that the proposed loss function is able to represent well the labels, but lacks of local capacity to distinguish well examples with some labels in common. We intend to reinforce the proposed loss by adding a term able to bring closer examples with most similar labels and repulse examples sharing few labels, i.e. adding a term based on example similarity. A second improvement concerns the optimization procedure which has shown instability. A more adapted gradient descent can be derived specifically for the considered hyperbolic representation.

# References

[BJK+15] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse Local Embeddings for Extreme Multi-label Classification. In *Advances in Neural Information Processing Systems 28*, pages 730–738. 2015.

| Dataset | | Hyp | SLEEC | Parabel | FastXML | PfastreXML | AnnexML |
|---|---|---|---|---|---|---|---|
| amazonCat | P@1 | 89.8% | 90.5% | 93.0% | 93.1% | 91.8% | 93.6% |
| | P@3 | 73.5% | 76.3% | 79.16% | 78.2% | 78.0% | 78.4% |
| | P@5 | 58.7% | 61.5% | 64.5% | 63.4% | 63.7% | 63.3% |
| wiki10 | P@1 | 85.6% | 85.9% | 84.3% | 83.0% | 83.6% | 86.5% |
| | P@3 | 72.9% | 73.0% | 72.6% | 67.5% | 68.6% | 74.3% |
| | P@5 | 62.4% | 62.7% | 63.4% | 57.8% | 59.1% | 64.2% |
| Delicious-200K | P@1 | 46.5% | 47.85% | 47.0% | 43.07% | 41.72% | 46.6% |
| | P@3 | 40.8% | 42.21% | 40.1% | 38.66% | 37.83% | 40.8% |
| | P@5 | 37.8% | 39.43% | 36.6% | 36.19% | 35.58% | 37.8% |
| WikiLSHTC-325K | P@1 | 43.2% | 54.8% | 65.0% | 49.8% | 56.0% | 63.4% |
| | P@3 | 26.5% | 33.4% | 43.2% | 33.1% | 36.8% | 40.7% |
| | P@5 | 19.4% | 23.9% | 32.1% | 24.5% | 27.1% | 29.8% |
| Amazon-670K | P@1 | 31.1% | 35.1% | 44.9% | 37.0% | 39.5% | 42.0% |
| | P@3 | 28.1% | 31.3% | 39.8% | 33.3% | 35.8% | 36.7% |
| | P@5 | 26.0% | 28.7% | 36.0% | 30.5% | 33.1% | 32.8% |

Table 3: Results of our approach (Hyp) compared to several state of the art approaches. For *SLEEC* and *AnnexML*, the predictions are made by aggregating the label vote of 15 different models. *FastXML* and *PfastreXML* aggregate 50 trees prediction. *AnnexML* merging 1 to 3 classifiers results. All reported results come from the *Extreme Classification repository*, except for *AnnexML* for which the reported results came from [Tag17].

| #Learners | # Dimensions | Hyperbolic | Euclidean |
|---|---|---|---|
| 1 | 5 | 78.9/54.1/43.1 | 77.02/53.5/42.5 |
| 3 | 5 | 81.3/60.0/50.0 | 81.1/59.2/48.7 |
| 6 | 5 | 81.1/61.5/52.0 | 81.3/60.3/50.4 |
| 9 | 5 | 81.2/61.8/52.6 | 81.2/60.6/50.7 |
| 12 | 5 | 81.2/61.8/53.0 | 81.2/60.6/50.9 |
| 15 | 5 | 81.2/61.6/53.1 | 81.2/60.6/50.9 |
| 1 | 10 | 81.5/61.6/50.9 | 79.1/61.4, 50.4 |
| 3 | 10 | 82.8/66.5/56.0 | 83.0/66.2/55.0 |
| 6 | 10 | 82.6/67.4/57.5 | 83.2/67.3/56.9 |
| 9 | 10 | 82.6/67.6/57.7 | 83.1/67.5/57.4 |
| 12 | 10 | 82.5/67.4/57.7 | 83.1/67.4/57.6 |
| 15 | 10 | 82.6/67.6/58.0 | 83.1/67.5/57.5 |
| 1 | 25 | 82.9/67.7/57.1 | 82.5/67.8/57.5 |
| 3 | 25 | 84.8/70.1/59.7 | 84.8/71.0/60.5 |
| 6 | 25 | 84.6/71.0/60.3 | 85.6/71.7/61.5 |
| 9 | 25 | 84.5/71.0/60.5 | 85.5/77.9/61.7 |
| 12 | 25 | 84.5/71.1/60.7 | 85.8/71.9/61.7 |
| 15 | 25 | 84.6/71.3/60.7 | 85.8/72.0/67.9 |

Table 4: Influence of the size of the representations and the number of aggregated models for the *Wiki10-31K* corpus for the precision @1,3,5. Hyperbolic refers to our approach, euclidean refers to an euclidean embedding similar to the AnnexML algorithm without pre-clustering.

| Dataset | # learners | P@1 | P@3 | P@5 | SLEEC (15 Learners) |
|---|---|---|---|---|---|
| AmazonCat | 1 | 86.7% | 69.1% | 54.5% | 90.5%/76.3%/61.5% |
| | 3 | 88.7% | 72.1% | 57.3% | |
| | 6 | 89.4% | 72.8% | 58.2% | |
| | 9 | 89.6% | 73.3% | 58.5% | |
| | 12 | 89.7% | 73.4% | 58.6% | |
| | 15 | 89.8% | 73.5% | 58.7% | |
| Wiki10-31k | 1 | 83.9% | 70.1% | 59.7% | 85.9%/73.0%/62.7% |
| | 3 | 85.3% | 72.3% | 61.6% | |
| | 6 | 85.7% | 72.6% | 62.1% | |
| | 9 | 85.6% | 72.8% | 62.4% | |
| | 12 | 85.5% | 73.0% | 62.5% | |
| | 15 | 85.6% | 72.9% | 62.4% | |
| Delicious Large 200K | 1 | 41.1% | 35.4% | 32.5% | 47.9%/42.2%/39.4% |
| | 3 | 44.6% | 39.0% | 36.1% | |
| | 6 | 45.8% | 40.2% | 37.2% | |
| | 9 | 46.3% | 40.6% | 37.6% | |
| | 12 | 46.5% | 40.8% | 37.8% | |
| | 15 | 46.5% | 40.8% | 37.8% | |
| WikiLSHTC-325K | 1 | 36.4% | 20.5% | 14.5% | 54.8%/33.4%/23.9% |
| | 3 | 40.9% | 24.1% | 17.3% | |
| | 6 | 42.4% | 25.5% | 18.5% | |
| | 9 | 43.0% | 26.2% | 19.1% | |
| | 12 | 43.2% | 26.5% | 19.4% | |
| Amazon-670K | 1 | 19.3% | 18.0% | 17.1% | 35.1%/31.3%/28.6% |
| | 3 | 23.9% | 21.8% | 20.3% | |
| | 6 | 27.4% | 24.8% | 23.04% | |
| | 9 | 29.2% | 26.4% | 24.5% | |
| | 12 | 30.3% | 27.4% | 25.4% | |
| | 15 | 31.1% | 28.1% | 26.0% | |

Table 5: Precision @k (1/3/5) depending on the number of aggregated models for our method and *SLEEC*.

| Label Name | Norm | Frequency |
|---|---|---|
| wikipedia | 0.6 | 80.7 |
| wiki | 6.8 | 41.9 |
| reference | 10.7 | 28.3 |
| history | 14.6 | 18.7 |
| science | 21.5 | 12.3 |
| research | 23.0 | 14.5 |

Table 6: Label sorted by the norm of their representation for the *wiki10* dataset.

[CL12] Yao-nan Chen and Hsuan-tien Lin. Feature-aware label space dimension reduction for multi-label classification. In *Advances in Neural Information Processing Systems 25*, pages 1529–1537, 2012.

[GBH18] Octavian Ganea, Gary Becigneul, and Thomas Hofmann. Hyperbolic Neural Networks. In *Advances in Neural Information Processing Systems 31*, pages 5345–5355. 2018.

[HKLZ09] Daniel J Hsu, Sham M Kakade, John Langford, and Tong Zhang. Multi-Label Prediction via Compressed Sensing. In *Advances in Neural Information Processing Systems 22*, pages 772–780, 2009.

[JPV16] Himanshu Jain, Yashoteja Prabhu, and Manik Varma. Extreme Multi-label Loss Functions for Recommendation, Tagging, Ranking & Other Missing Label Applications. In *Proceedings of the 22nd ACM SIGKDD*, pages 935–944, 2016.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[NK17] Maximillian Nickel and Douwe Kiela. Poincaré Embeddings for Learning Hierarchical Representations. In *Advances in Neural Information Processing Systems 30*, pages 6338–6347. 2017.

[PKH+18] Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. Parabel: Partitioned Label Trees for Extreme Classification with Application to Dynamic Search Advertising. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 993–1002, 2018.

[PV14] Yashoteja Prabhu and Manik Varma. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 263–272, 2014.

[SDSGR18] Frederic Sala, Chris De Sa, Albert Gu, and Christopher Re. Representation tradeoffs for hyperbolic embeddings. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 4460–4469, 10–15 Jul 2018.

[Tag17] Yukihiro Tagami. Annexml: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD*, pages 455–464, 2017.

[TL12] Farbound Tai and Hsuan-Tien Lin. Multilabel classification with principal label space transformation. *Neural Comput.*, 24(9):2508–2542, September 2012.

[TLH17] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. Enabling efficient question answer retrieval via hyperbolic neural networks. *CoRR*, abs/1707.07847, 2017.

[YHD+17] Ian E.H. Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. PPDsparse: A Parallel Primal-Dual Sparse Method for Extreme Classification. In *Proceedings of the 23rd ACM SIGKDD*, pages 545–553, 2017.

[YHZ+] Ian E H Yen, Xiangru Huang, Kai Zhong, Pradeep Ravikumar, and Inderjit S Dhillon. PD-Sparse : A Primal and Dual Sparse Approach to Extreme Multiclass and Multilabel Classification. page 11.

[ZH05] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.

[ZLLG18] Min-Ling Zhang, Yu-Kun Li, Xu-Ying Liu, and Xin Geng. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science*, pages 1–12, 2018.