Report on Hyperpolic Community detection

September 29, 2019

1 Introduction

We plan to adapt the paper "Learning Community Embedding with Community Detection and Node Embedding on Graphs" using Hyperbolic space instead of euclidean one. Today community graph embedding rely on projecting data from a graph (such as adjacency matrix) on an continuous space, and the retrieve community by clustering based algorithms. The projection often involve a small representation space, moreover severall communauties are overlapping each others. In particular hierarchical community can exists, taking it into account can be relevant. If mainly used representation space is euclidean, some others manifolds can be used. Particularly hyperbolic manifolds have prooven their efficiency to embed hierarchical data and performs well in low dimenssion space. In this document we report results and experiments in progress experiments on community detection using hyperbolic riemnanian manifolds.

2 Learning Community Embedding with Community Detection an Node Embedding Graph

Before describing our approach we sum up the main related articles. The paper propose to face Community detection in jointly learning detection process and embedding projection function.

2.1 Community Detection embedding

In order to retrieve community, authors propose to associate for each examples embeddings a community probability by $p(z_i = k)$ the probability of node v_i to belong to the community indexed by k. They model the prosteriot $p(v_i|z, \Theta)$ by a multivariate gaussian distritbution, thus we can write :

$$p(v_i|z_i = k; \phi_i, \psi_k, \Sigma_k) = \mathcal{N}(\phi_i|\psi_k, \Sigma_k)$$

With ψ_k, Σ_k respectively the mean and variance of the gaussian distribution.

2.2 Node Embedding

The embeddings are obtained by rapproaching neigbhors in terms of graph distance, the first cost function associate is :

$$O_1 = -\alpha_1 \sum_{(v_i,v_j)\in E} log(\sigma(z_i^t z_j)))$$

with E the vertices. Unfortunatelly nodes at distance one are not always in the same community and nodes at a further distance can be in a same community. Thus the *DeepWalk* algorithm propose to perform a random walk over the graph allowing to find neigbhors. The neigbhorhood of a node v_i is called context of v_i, denoted C_i:

$$O_2 = -\alpha_2 \sum_{v_i\in V} \sum_{v_j\in C_i} \left[log(\sigma(z_i^t z_j))) + \sum_{t=1, v_l\notin C_i}^{T} log(\sigma(z_i^t z_l)) \right]$$

Once the model is fit we must ensure that it fit the multivariate gaussian prior. To connect both, the node embedding must take into account detection process, thus they add an additional loss fixing all except the embedding. This is given by :

$$O_3 = -\alpha_3 \sum_{v_i\in V} log\left[\sum_{k=1}^{K} p(z_i = k) p(v_i|z_i = k; \phi_i, \psi_k, \Sigma_k) \right]$$

3 Adapting to Hyperbolic

3.1 Node Embedding without a priori on Community detection

In this section we plan to find an alternative to O_1 and O_2 loss function because we can not use O_1 and O_2 without considering the poincaré ball distance. To keep the same meaning we can modelize the propabilities of users being in the same community $P((v_i, v_j) \in C|z_i, z_j)$ we can thus simply modelise it by an exponential distribution :

$$p((v_i, v_j) \in E|z_i, z_j) = \lambda e^{-\lambda - d_h(z_i, z_j)}$$

with d_h the distance associate to the hyperbolic space. Thus we can use the following loss function considering that items sharing edge often are in the same community :

$$O_{h,1} = -\alpha_1 \sum_{(v_i,v_j)\ \in E} log(p((v_i, v_j) \in C|z_i, z_j))$$

With λ a parameter that may be selected by grid search, however we curently use in experiments $\lambda = 1$.

Dimension	performance hyperbolic	performance euclidean
2	74.6 ± 2.9	69.9 ± 4.5
3	81.5 ± 3.9	79.2 ± 3.2
4	82.9 ± 3.6	84.4 ± 2.2
5	86.2 ± 2.5	86.8 ± 3.0
10	88.6 ± 2.1	87.7 ± 2.9

Table 1: KMeans football 10 kmeans init

Dimension	performance hyperbolic	performance euclidean
2	93.2 ± 1.4	85.8 ± 17.3
3	96.1 ± 1.4	92.9 ± 1.5
4	91.7 ± 7.4	94.1 ± 0.0
5	94.1 ± 0.0	94.1 ± 0.0
10	94.1 ± 0.0	94.1 ± 0.0

Table 2: KMeans karate 10 kmeans init

We also need to adapt a loss using negative sampling and randomWalks

$$O_{h,2} = -\alpha_2 \sum_{(v_i,v_j)\in R} log\left[\frac{p((v_i,v_j)\in C|z_i,z_j)}{\sum\limits_{v_k\in N_i\cup v_j} p((v_i,v_k)\in E|z_i,z_k)}\right]$$

$$= \alpha_2 \sum_{(v_i,v_j)\in R} log\left[1 + \sum_{v_k\in N_i\cup v_j} \lambda e^{-\lambda(d_h(z_i,z_j)-d_h(z_i,z_k))}\right]$$

For optimization we use gradient descent with retractation similarly to the Nickel et al paper [**?**]. We may also use the gradient descent proposed by Wilson and Leimeister [**?**], showing better convergence on severall tasks, similarly we can also use the lorentz model method Kiala et al [**?**] and then projecting in the poincaré model.

Considering $f_\theta : x \to r$ the projection function and $L(x, y)$ the loss function, the Nickel et al methods update the parameters θ by :

$$\theta_{t+1} = \theta_t - \alpha\frac{(1-||\theta_t||^2)^2}{4}\Delta_E$$

With Δ_E the gradient of loss function with respect to θ_t

The second method proposed is based on optimization on the tangent space and then using exponential map to remap gradient on the hyperbolic space.

3.2 EM

3.3 Connecting both

To connect both embeddings must fit the prior, This is done by minimizing :

$$O_{h,3} = -\alpha_3 \sum_{v_i\in V} log\left[\pi_{ik}\sum_{k=0}^{K}\frac{1}{\zeta(i)}e^{\frac{-(d_h(x,\mu_k))^2}{2\sigma_k^2}}\right]$$

$$O'_{h,3} = -\alpha_3 \sum_{v_i\in V}\sum_{k=0}^{K}\pi_{ik} log\left[\frac{1}{\zeta(i)}e^{\frac{-(d_h(x,\mu_k))^2}{2\sigma_k^2}}\right]$$

$$O_{h,3} \leq O'_{h,3}$$

4 Evaluating EM Algorithm On large dataset

5 K-Means results

Dimension	performance hyperbolic	performance euclidean
2	93.2 ± 1.4	85.8 ± 17.3
3	96.1 ± 1.4	92.9 ± 1.5
4	91.7 ± 7.4	94.1 ± 0.0
5	94.1 ± 0.0	94.1 ± 0.0
10	94.1 ± 0.0	94.1 ± 0.0

Table 3: KMeans BOOKS 10 kmeans init

Dimension	performance hyperbolic	performance euclidean
2	72.7 ± 2.1	68.9 ± 5.7
3	71.3 ± 6.8	74.9 ± 6.9
4	74.5 ± 7.8	77.6 ± 5.0
5	75.3 ± 8.4	79.2 ± 0.6
10	78.6 ± 6.6	84.2 ± 2.1

Table 5: KMeans dblp 10 kmeans init