HierarX : un outil pour la découverte de hiérarchies dans des espaces hyperboliques à partir de similarités

François Torregrossa*,**, Guillaume Gravier**
Vincent Claveau**, Nihel Kooli*

*Solocal, Rennes 35000
{ftorregrossa,nkooli}@solocal.com,
http://www.solocal.com
**IRISA, Rennes 35000
francois.torregrossa@irisa.fr
guillaume.gravier@irisa.fr
vincent.claveau@irisa.fr
http://www.irisa.fr

Résumé. Cet article introduit l'outil HierarX, permettant de projeter des données dans des espaces hyperboliques : Lorentz ou Poincaré. À partir de similarités entre des mots ou de leur représentation dans des espaces de grandes dimensions, HierarX incorpore des connaissances dans des géométries hyperboliques de petites dimensions. Ces dernières ont la particularité de représenter l'information sous forme de hiérarchies continues. Ce travail présente le fonctionnement de HierarX ainsi que ses principaux cadres d'utilisation.

1 Introduction

Les plongements de mots ou *word embeddings* sont des méthodes bien connues et largement utilisées pour la compréhension du langage naturel. À partir de diverses sources, ces solutions proposent de calculer la représentation continue des mots ou concepts composant le langage. En finalité, des relations géométriques, utiles aux systèmes automatiques, en émergent.

En pratique, les sources de données sont multiples : des phrases brutes ou encore des similarités. Word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) et FastText (Bojanowski et al., 2017) sont par exemple des méthodes travaillant sur des phrases brutes en vue de découvrir des relations paradigmatiques ou syntagmatiques. D'autres méthodes étudient les similarités entre paires de mots, par exemple les travaux de Sun et al. (2015) et plus récemment ceux de Nickel et Kiela (2018). Leur proposition est de reconstruire des données hiérarchiques en utilisant les similarités entre éléments, qui forment une source de données plate et déstructurée. Malgré des résultats prometteurs, les implémentations des deux solutions n'ont pas été rendues publiques. De plus, celles-ci sont particulièrement difficiles à réaliser en pratique, puisqu'elles font intervenir des espaces hyperboliques, plus adaptés à la représentation des hiérarchies de façon continue. HierarX est le logiciel que nous proposons pour combler ce manque.

Dans ce travail nous expliquons l'intérêt de la géométrie hyperbolique ainsi que les logiciels s'approchant de HierarX en section 2. Par la suite, nous évoquons le fonctionnement de HierarX en section 3. La section 4 présente les usages du logiciel ainsi que les résultats obtenus sur diverses sources de données. La dernière section conclue la présentation.

2 État de l'art

La géométrie euclidienne est généralement utilisée pour les plongements de mots du fait qu'elle soit simple à appréhender et requiert peu de calcul. Elle consiste à dire que le plus court chemin entre deux points est toujours une ligne droite. D'autres axiomes la complètent et il en résulte de nombreuses propriétés telles que, par exemple, l'inégalité triangulaire. Ces dernières ont des aspects vertueux pour incorporer des paradigmes via des représentations géométriques. En revanche, elles peuvent contraindre la capacité à projeter d'autres relations, comme des relations hiérarchiques. Sun et al. (2015) expliquent précisément les problématiques soulevées par les caractéristiques intrinsèques à la géométrie d'Euclide. On notera principalement la problématique de limite du nombre maximum de plus proches voisins. En effet, dans un espace euclidien, un point peut avoir au plus un certain nombre de plus proches voisins. Cette contrainte empêche la formation de hiérarchies continues dans l'espace, puisqu'on ne peut pas avoir un élément racine qui serait le plus proche voisin d'une infinité d'autres points.

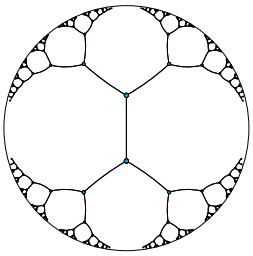


FIG. 1: Projection d'un arbre binaire dans le disque de Poincaré (source image : Nickel et Kiela (2017)).

Sun et al. (2015) ont décidé d'investiguer un espace hyperbolique (Minkowski) dans l'optique de contrer ces contraintes euclidiennes, et ainsi faciliter la formation de hiérarchies continues. Par la suite, Nickel et Kiela (2017) ont démontré l'efficacité de l'espace hyperbolique de Poincaré pour l'incorporation continue de base de connaissances comme WordNet (Miller, 1995). Cette efficacité reposant notamment sur la facilité d'encoder un arbre binaire dans un espace hyperbolique comme celui de Poincaré. La Figure 1 en est un exemple. Pour ce faire, il suffirait de placer au centre les éléments racines de l'arbre, et aux abords du disque ses feuilles. Par la suite, il faut utiliser la distance hyperbolique pour positionner le reste des noeuds dans le disque, de façon cohérente.

Plus tard, les mêmes auteurs démontrent (Nickel et Kiela, 2018) l'intérêt d'un autre espace hyperbolique, celui de Lorentz. FastText et GloVe ont aussi été implémentés dans des espaces hyperboliques pour tenter de découvrir des hiérarchies à partir de textes bruts, sans succès significatif (Leimeister et Wilson, 2018; Tifrea et al., 2018). Néanmoins, ces travaux font émerger plusieurs points importants :

- Sur des tâches de similarité (Finkelstein et al., 2002; Vulić et al., 2017; Hill et al., 2015), les espaces hyperboliques ont des performances remarquables et significativement meilleures en petites dimensions, en comparaison avec leurs équivalents euclidiens.

- Les espaces hyperboliques montrent des capacités à incorporer des bases de connaissances très bonnes, contrairement aux espaces euclidiens qui sont très mal adaptés à cela (Nickel et Kiela, 2017, 2018).
- Les espaces hyperboliques semblent faire émerger des hiérarchies à partir de similarités à plat en suivant Sun et al. (2015) et Nickel et Kiela (2018).

Dans l'état de l'art, il existe plusieurs implémentations notables des espaces hyperboliques. Par exemple, celle de Nickel et Kiela (2017) [1] qui propose l'incorporation d'une base de connaissances dans Poincaré ou Lorentz. En revanche, le projet ne permet pas de faire émerger des hiérarchies continues à partir de similarités à plat. Un autre projet remarquable est celui de Kochurov et al. (2019) qui se base sur la librairie PyTorch (Paszke et al., 2017). Ce projet fait l'implémentation des différents espaces hyperboliques et donne la possibilité de réaliser toutes sortes de tâches dans ces espaces. Enfin, Ganea et al. (2018) introduisent les réseaux de neurones hyperboliques ce qui pourrait donner la possibilité d'exploiter les plongements hyperboliques.

HierarX est donc motivé par la volonté de fournir un logiciel très rapide permettant de construire des hiérarchies continues à partir de similarités entre des éléments divers. Comme évoqué, cette opération n'est pas réalisable simplement et rapidement avec les implémentations de l'état de l'art. Or, cela est très utile pour la reconstruction de hiérarchies depuis n'importe quelle source de données. De plus, l'efficacité des espaces hyperboliques est démontrée sur plusieurs tâches.

3 Fonctionnement de HierarX

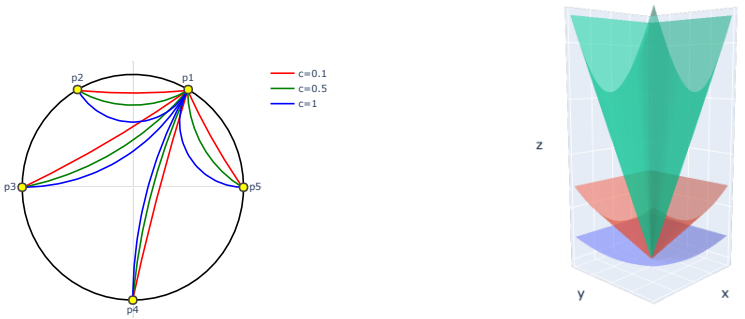
Cette section détaille les implémentations des espaces hyperboliques ainsi que le processus d'optimisation.

3.1 Espaces Hyperboliques

Les espaces de Lorentz et de Poincaré sont présents dans HierarX. Chaque espace présente des avantages et inconvénients comme présentés par Nickel et Kiela (2018). En effet, dans celui de Lorentz, les calculs restent simples ce qui rend l'optimisation plus rapide. Cependant, son interprétation visuelle est moins claire que celle dans l'espace de Poincaré. À l'inverse, ce dernier est plus complexe et requiert davantage de calculs, ce qui ralentit l'optimisation. Il est en revanche plus stable, car l'espace de Poincaré est défini comme une boule ouverte alors que celui de Lorentz est donné comme un hyperplan. Ce dernier point n'est pas négligeable puisqu'il permet la parallélisation sécurisée de l'optimisation chez Poincaré, ce qui n'est pas possible chez Lorentz. Il est important de noter qu'il est possible de passer de l'un à l'autre en suivant la fonction définie dans le travail de Nickel et Kiela (2018).

Les espaces de Poincaré sont implémentés en suivant les définitions proposées dans Ganea et al. (2018). Nous reprenons également le paramètre de célérité c qui permet le contrôle de la courbure de l'espace, comme montré par la Figure 2a. Lorsque c tend vers zéro, les géodésiques tendent à devenir des lignes droites : on revient à la géométrie euclidienne.

1. https://github.com/facebookresearch/poincare-embeddings



(a) Géodésiques dans la boule de Poincaré en dimension 2 représentée par le cercle. Le paramètre c contrôle leur courbure.

(b) Représentation des espaces de Lorentz, en partant du bas vers le haut : $\mathcal{L}^2_{0.1}$, $\mathcal{L}^2_1$ et $\mathcal{L}^2_{10}$

FIG. 2: Influence de la célérité dans les espaces hyperboliques de Lorentz et Poincaré.

Pour l'espace de Lorentz, nous suivons les définitions de Nickel et Kiela (2018). Nous proposons dans HierarX de généraliser le paramètre c qui ajuste la courbure dans Poincaré à Lorentz. Pour ce faire nous redéfinissons l'espace : $\mathcal{L}^n_c = \{x \in \mathbb{R}^{n+1} \mid \langle x, x\rangle_{\mathcal{L}^n_c} = -1\}$, avec $\langle x, y\rangle_{\mathcal{L}^n_c} = -x_0 y_0 + c\sum_{i=1}^n x_i y_i$. Cette généralisation a des propriétés comparables à la célérité dans Poincaré et préserve la bilinéarité du produit scalaire de Lorentz. Comme présenté en Figure 2b, lorsque c tend vers 0, l'espace de Lorentz $\mathcal{L}^2_{c\to 0}$ semble s'identifier à $\mathbb{R}^2$. Lorsque c devient grand, la courbure de l'espace s'intensifie.

3.2 Optimisation

Le processus d'optimisation repose sur des similarités entre des paires de mots. Elles peuvent être fournies via un embedding de type FastText (au format GloVe) ou alors via une matrice de similarités. Dans le cas d'un embedding, la similarité est déduite du produit scalaire entre les vecteurs. Pour bénéficier de la similarité cosinus, les vecteurs doivent être normalisés.

Nous reprenons la méthode proposée par Nickel et Kiela (2018) pour incorporer des similarités dans des espaces hyperboliques. Nous notons $s(e_1, e_2)$ la similarité entre deux éléments e_1 et e_2 d'un vocabulaire V. Chaque élément e est représenté par un vecteur v_e dans l'espace hyperbolique, et on note $\Theta = (v_e)_{e\in V}$ l'ensemble de ces vecteurs. Par la suite, un élément cible e_t est tiré aléatoirement dans le vocabulaire. Avec celui-ci, on tire m autres éléments $(e_i)_{i\in[\![1,m]\!]}$. Le but de la procédure est de faire en sorte que l'élément qui a la plus forte similarité avec e_t, parmi ces m, soit le plus proche de e_t dans la projection en terme de distance hyperbolique d. Notant $i_{\max} = \operatorname{argmax}_{i\in[\![1,m]\!]} s(e_t, e_i)$, l'indice de l'élément maximisant la similarité avec la cible dans le lot, cela revient à vouloir maximiser, selon Nickel et Kiela (2018) :

$$P(\operatorname*{argmin}_{i\in[\![1,m]\!]} d(v_{e_t}, v_{e_i}) = i_{\max}|\Theta) = \frac{e^{-d(v_{e_t}, v_{e_{i_{\max}}})}}{\sum_{i=1}^m e^{-d(v_{e_t}, v_{e_i})}}.$$

Pour plus de simplicité, nous préférons minimiser le problème équivalent suivant :

```
5 2
hierarx 0.37961 0.52684
learns 0.25774 0.52448
hyperbolic 0.09547 0.51548
word 0.093367 0.47123
representations 0.1313 1.0844
```

(a) Format GloVe (Pennington et al., 2014) pour l'entrée embedding.

```
Left,signals,3.395
coaches,Todd,4.502
Todd,aware,2.446
aware,signals,4.765
bridges,kilometers,7.450
intellectual,fundamental,7.773
```

(b) Format CSV pour l'entrée matrice de similarité.

FIG. 3: Exemples des deux formats possibles en entrée de HierarX.

$$L(e_t, (e_i)_{i\in[\![1,m]\!]}, \Theta) = \log(1 + e^{d(v_{e_t}, v_{e_{i_{\max}}})}) + \sum_{i=1, i\neq i_{\max}}^{m} \log(1 + e^{-d(v_{e_t}, v_{e_i})}).$$

Quand le vocabulaire est large, nous ajoutons artificiellement des éléments proches de la cible, au sens de la similarité. Ceci dans le but d'avoir un échantillonnage qui représente mieux le voisinage du mot cible.

Par la suite, la minimisation se fait par la méthode de la descente du gradient stochastique en espace Riemannien, en se basant sur l'implémentation de geoopt (Kochurov et al., 2019) ainsi que plusieurs autres articles (Nickel et Kiela, 2018, 2017; Bécigneul et Ganea, 2018; Tifrea et al., 2018). Elle repose sur le calcul des gradients euclidiens, leur projection dans l'espace hyperbolique choisi puis l'application de ces dernières aux paramètres du modèle.

3.3 Projection avec HierarX

HierarX est disponible en ligne [2] et permet, via un terminal, d'appliquer le processus d'optimisation décrit en section 3.2 dans un des espaces de la section 3.1 à un ensemble de similarités fourni via un plongement euclidien ou une matrice. Un ensemble d'arguments contrôle le déroulement de l'entraînement ce qui produit des plongements hyperboliques avec différentes propriétés, nous en présentons en section 4.

Il y a deux formats possibles en entrée de HierarX correspondant aux deux entrées prises en charge : un embedding ou une matrice de similarités. Des exemples sont présentés en Figure 3. Pour l'embedding, nous reprenons le format standard GloVe : la première ligne décrit le nombre de mots ainsi que la dimension, et les lignes suivantes contiennent un mot suivi de ses coordonnées dans la représentation. Pour la matrice, nous prenons un format csv de trois colonnes : mot1,mot2,s(mot1, mot2). Les deux premières colonnes détaillent la paire de mots et la dernière la similarité entre les mots précédents. Ce format est amené à évoluer.

Le logiciel requiert un répertoire d'expérience dans lequel est sauvegardé l'état de l'apprentissage : l'embedding hyperbolique ainsi que d'autres éléments utiles. À la fin de la phase d'optimisation, il faudra convertir cet embedding (sauvegardé en format binaire) dans un format GloVe (similaire à celui passé en entrée). Pour ce faire, HierarX propose un convertisseur intégré permettant d'effectuer l'opération. Davantage d'informations sont détaillées dans le readme du projet.

2. https://github.com/pagesjaunes/HierarX

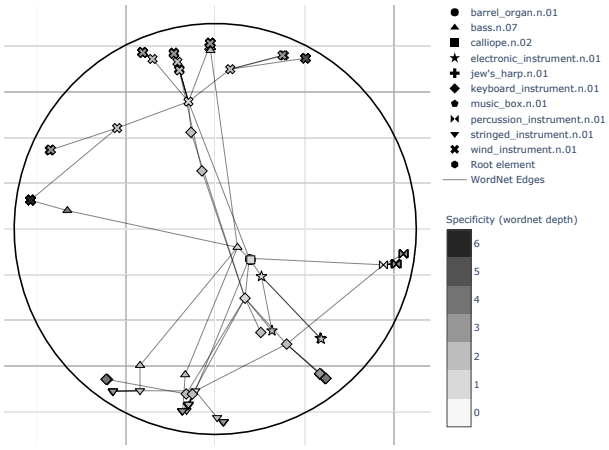


FIG. 4: Projection dans l'espace de Poincaré (dimension 2) des concepts appartenant aux instruments de musique dans la source de données WordNet. Le résultat est obtenu en fournissant à HierarX une matrice de similarités où la similarité entre deux concepts se base sur le plus court chemin les reliant dans WordNet.

4 Usages et résultats

Nous proposons de mettre en valeur les deux principaux cadres d'utilisation du logiciel, à savoir : la découverte de hiérarchies continues à partir de similarités à plat et la compression d'une réprésentation euclidienne dans une représentation hyperbolique. De multiples jeux de données (Priebe et al., 2006; Bouckaert et al., 2012), non présentés ici, ont été testés pour vérifier la ressemblance entre notre implémentation avec les résultats présentés dans Nickel et Kiela (2018).

4.1 Découverte de hiérarchies

Nous proposons de faire la découverte de hiérarchies continues à partir de similarités fournies par WordNet. Pour ce faire, une sous-famille de concepts est extraite du graphe et la similarité entre deux concepts se base sur le plus court chemin les reliant dans WordNet. Nous constituons alors une matrice de similarités entre tous les concepts de cette sous-famille. La Figure 4 est une projection obtenue en utilisant HierarX sur la matrice de similarités de la famille des instruments de musique. On peut remarquer que les concepts s'organisent dans l'espace en fonction de leur catégorie. Ainsi, les instruments à cordes, à vent et à percussions (ou clavier) se répartissent dans trois zones différentes de l'espace. Il est aussi important de noter l'encodage de la spécificité dans la norme des vecteurs. En effet, la corrélation de spearman entre la norme et la profondeur du concept dans WordNet vaut 0.64. Cela signifie que les concepts les plus spécifiques vont se retrouver aux abords de la boule de Poincaré. À l'inverse les plus génériques se situent au centre (comme le concept racine, par exemple).

Embedding	HyperLex	WordSim
FastText (300 dimensions)	0.20	0.75
Poincaré (20 dimensions)	0.28	0.38

TAB. 1: Résultats avant/après la compression d'un embedding FastText de 300 dimensions dans un espace de Poincaré à 20 dimensions. Le coefficient de Spearman est reporté pour chaque corpus (les pvaleurs sont très faibles). Le vocabulaire est composé des 100000 mots les plus fréquents. Pour HyperLex, le score is_a défini par Nickel et Kiela (2017) est utilisé pour l'espace hyperbolique. Pour WordSim, la distance hyperbolique est choisie. Pour FastText, la similarité cosinus sert pour le calcul des similarités dans les deux cas.

4.2 Réduction de dimensions

La seconde utilisation de HierarX fait intervenir les plongements continus de mots comme FastText. Le but est d'obtenir des hiérarchies continues à partir des similarités entre les mots acquises par l'apprentissage de plongements sur du texte brut. Cela permet aussi de réduire la dimension des espaces euclidiens. La Table 1 donne les performances du plongement et de sa compression sur diverses tâches : HyperLex (Vulić et al., 2017), et WordSim (Finkelstein et al., 2002). La première tâche évalue la capacité à déterminer si un concept est un hyponyme ou hyperonyme d'un autre, alors que la seconde évalue la similarité sémantique. On constate une perte sur la performance au profit d'une représentation moins lourde pour WordSim. En revanche, le score est légèrement plus haut pour la tâche avec HyperLex. Cela penche en faveur de la capacité des représentations hyperboliques à découvrir des hiérarchies. D'autres hyperparamètres peuvent donner des représentations qui sont meilleures sur WordSim et moins bonnes sur HyperLex. Il faut donc les sélectionner en fonction du besoin.

5 Conclusion

Pour conclure, HierarX est un logiciel disponible en ligne [3] et en développement continu. Nous proposons cette implémentation dans l'objectif de fournir un outil manquant à la communauté. En effet, il utilise des similarités pour construire des représentations hyperboliques plus adaptées à la représentation de hiérarchies continues. Deux exemples sont proposés dans cet article. L'exploitation de ces représentations est encore un sujet à explorer.

Références

Bécigneul, G. et O. Ganea (2018). Riemannian adaptive optimization methods. *CoRR abs/1810.00760*.

Bojanowski, P., E. Grave, A. Joulin, et T. Mikolov (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*.

3. https://github.com/pagesjaunes/HierarX

Bouckaert, R., P. Lemey, M. Dunn, S. Greenhill, A. Alekseyenko, A. Drummond, R. Gray, M. Suchard, et Q. Atkinson (2012). Mapping the origins and expansion of the indo-european language family. *Science 337*(6097), 957–960.

Finkelstein, L., E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, et E. Ruppin (2002). Placing search in context : The concept revisited. *ACM Transactions on Information Systems 20*(1), 116–131.

Ganea, O., G. Bécigneul, et T. Hofmann (2018). Hyperbolic neural networks. *CoRR abs/1805.09112*.

Hill, F., R. Reichart, et A. Korhonen (2015). SimLex-999 : Evaluating semantic models with (genuine) similarity estimation. *American Journal of Computational Linguistics 41*.

Kochurov, M., S. Kozlukov, R. Karimov, et V. Yanush (2019). Geoopt : Adaptive riemannian optimization in pytorch. `https://github.com/geoopt/geoopt`.

Leimeister, M. et B. J. Wilson (2018). Skip-gram word embeddings in hyperbolic space. *CoRR abs/1809.01498*.

Mikolov, T., K. Chen, G. Corrado, et J. Dean (2013). Efficient estimation of word representations in vector space. In *ICLR 2013, Workshop Track Proceedings*.

Miller, G. A. (1995). Wordnet : A lexical database for english. *Commun. ACM 38*(11), 39–41.

Nickel, M. et D. Kiela (2017). Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems 30*, pp. 6338–6347.

Nickel, M. et D. Kiela (2018). Learning continuous hierarchies in the lorentz model of hyperbolic geometry. *CoRR abs/1806.03417*.

Paszke, A., S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, et A. Lerer (2017). Automatic differentiation in pytorch. In *NIPS-W*.

Pennington, J., R. Socher, et C. D. Manning (2014). Glove : Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

Priebe, C. E., J. M. Conroy, D. J. Marchette, et Y. Park (2006). Enron data set. `http://cis.jhu.edu/~parky/Enron/enron.html`.

Sun, K., J. Wang, A. Kalousis, et S. Marchand-Maillet (2015). Space-time local embeddings. In *Advances in Neural Information Processing Systems 28*, pp. 100–108.

Tifrea, A., G. Bécigneul, et O. Ganea (2018). Poincaré glove : Hyperbolic word embeddings. *CoRR abs/1810.06546*.

Vulić, I., D. Gerz, D. Kiela, F. Hill, et A. Korhonen (2017). HyperLex : A large-scale evaluation of graded lexical entailment. *American Journal of Computational Linguistics 43*.

Summary

This article introduces the HierarX tool which projects multiple datasources into hyperbolic manifolds : Lorentz or Poincaré. From similarities between word pairs or continuous word representations in high dimensional spaces, HierarX is able to embed knowledge in hyperbolic geometries with small dimensionality. Those shape information into continuous hierarchies. This work presents the HierarX workflow as well as its main use-cases.