Going full hyper: hyperbolic and hypercomplex graph embeddings for ADMET modeling

Tuan Le*, **Marco Bertolini***, **Marc A. Boef***, **Floriane Montanari & Djork-Arné Clevert**
Machine Learning Research, Bayer AG,
13353 Berlin, Germany
{tuan.le2,marco.bertolini,marcarne.boef}@bayer.com
{floriane.montanari,djork-arne.clevert}@bayer.com

Abstract

We apply multitask learning in hyperbolic and hypercomplex spaces for predicting physico-chemical ADMET endpoints of small molecules. Our graph neural networks implementations show an increased overall predicting performance with respect to Euclidean-based methods. The performance gain of the quaternion model is especially accentuated in tasks with fewer data, strengthening the scope of multitask learning. In the hyperbolic approach, we experimentally observe that the network is making use of higher curvatures mainly in deeper layers, prompting us to explore hybrid networks, in which different layer geometries are combined.

1 Introduction

Quantitative structure-property relationship (QSPR) machine learning approaches have become a standard tool for the computational chemist in the quest for a more efficient and better targeted approach in the drug discovery process. Accurate in silico predictions of a candidate compound's physicochemical properties are crucial both in the design process to limit the risk of late-stage attrition, and in the screening phase by providing prioritizing criteria for compound synthesis.

While several recent works [1, 2, 3, 4] showed that Deep Learning models on graphs for property prediction are able to outperform the majority of the traditional machine learning approaches [5], their applicability has been restricted for the most part to predicting molecular properties for which enough data is available. In the context of absorption, distribution, metabolism, excretion, and toxicity (ADMET) properties, the generation of data is costly and subject to specific needs: different endpoints are measured at different stages in the project, resulting in sparse and inhomogeneous datasets, which are challenging to leverage for generating good predictions over several assays. A mitigation strategy to overcome this hurdle is multitask learning [6, 7], where neural networks must extract suitable data representations for solving several tasks at once. This approach effectively addresses the endpoints imbalance in the data by leveraging the correlation between the various endpoints [8]: data scarcity in a task is compensated by data abundance in another correlated task. Graph convolutional networks (GCNs), where each convolutional layer learns filters that are applied to a graph representation of the compound, have shown remarkable performance in learning and predicting molecule-based features. Combining these two approaches, [9, 10] show that extending multitask learning to the GCN realm led to the current state-of-the-art in predicting ADMET properties.

While we are still far from a comprehensive geometric approach to neural networks [11], deep learning on hyperbolic space has already shown encouraging results [12]. The most attention has been paid to hyperbolic manifolds with constant (negative) curvature, whose structure is sufficiently simple for being amenable to implementation, yet rich enough to show significant improvement in

*These authors contributed equally.

Machine Learning for Molecules Workshop at NeurIPS 2020. https://ml4molecules.github.io

performance over several – of most interest to us – graph-based tasks [13, 14]. Motivated by these results, we take a first step in exploring whether hyperbolic learning can be an extra asset in the application of deep learning in cheminformatics. We find that our hyperbolic models show solid performance with respect to the state-of-the-art GCN model [9]. In particular, the hybrid model, where both Euclidean and Hyperbolic convolutional layers are present, outperforms the GCN baseline in all tasks. The full Hyperbolic model has instead a slighly lower performance with respect to the hybrid counterpart: we hypothesize that given the relatively small size of the graphs in our ADMET dataset, the advantage introduced by the geometrical features is partially compensated by a more laborious training [15].

Geometry is only one of the possible generalizations concerning the structure of the embedding learned by a neural network, another being its underlying vector space structure. The quaternions define a (non-commutative) four-dimensional algebra $\mathbb{H}$ over the real numbers and they constitute the simplest hypercomplex number system. The characteristics of quaternion neural networks (QNNs) [16, 17, 18, 19] align surprisingly well with the advantages of multitask learning: first, a quaternion embedding is four times as rich as its Euclidean version, increasing the model capacity; additionally, the learned weights are shared within the quaternion representations, increasing and diversifying the amount of regularization. In this work, we observe that quaternion graph neural networks (QGNNs) [20] increase the performance of QSPR models. We report an R^2 score performance increase up to 84% on individual tasks, as well as a higher R^2 score average performance ($+0.06$) across all tasks.

2 Methods

Graph hyperrepresentation learning. We briefly describe the setting of our graph representation task. For our purposes a graph $\mathcal{G}$ is defined by the triple $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \{\mathbf{x}_v\}_{v\in\mathcal{V}})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges, and $\mathbf{x}_v$ represents the $M-$dimensional feature vector of node $v \in \mathcal{V}$. The objective of graph representation learning consists in constructing node representations $\mathbf{h}_v^{(l)} \in \mathcal{M}^l$ for each hidden layer l, where $\mathcal{M}^l$ is a manifold whose real dimension is m and $\mathbf{h}_v^{(0)} \equiv \mathbf{x}_v$.
Hidden vector representations of each node $v \in \mathcal{V}$ are obtained by iteratively aggregating and transforming the vector representations of its neighbors $\mathcal{N}_v$. After a number of L-iterations a pooling function, referred to as READOUT in the context of GNNs, is applied to obtain the final vector representation $\mathbf{h}_G$ of the entire graph. Summarizing, given a graph $\mathcal{G}$ we formulate GNNs as follows:

$$\mathbf{h}_v^{(l)} = \text{AGGR}\left(\left\{\mathbf{h}_u^{(l-1)}\right\}_{u\in\mathcal{N}_v\cup\{v\}}\right), \qquad \mathbf{h}_G = \text{READOUT}\left(\left\{\mathbf{h}_v^{(l)}\right\}_{v\in\mathcal{V}}^{l=0,1,..,L}\right). \tag{1}$$

Different GNNs are therefore defined by the specific implementation of (1).

In this work we explore two choices for our learning manifold: (i) $\mathcal{M} = \mathcal{H}_C^d$, the d-dimensional hyperboloid manifold of constant curvature $K = -C^{-1} < 0$, and (ii) $\mathcal{M} = \mathbb{H}$, the (flat) space of the algebra of quaternions. There are potential issues in defining learning on an arbitrary manifold $\mathcal{M}$. First, given $x, y \in \mathcal{M}$, operations such as product $x \otimes_{\mathcal{M}} y$ and sum $x \oplus_{\mathcal{M}} y$, which are at the core of defining feature transformations in neural networks, are often non-trivial or ill-defined, as $\mathcal{M}$ is not ensured to possess the structure of a vector space. Second, the non-linear activation function σ might not be manifold-preserving, that is, $\sigma : \mathcal{M} \to \mathcal{M}$ might not hold.

In hyperbolic space we overcome these obstacles by leveraging the vector space structure of the tangent space $T_x\mathcal{H}_C^d$ at a point $x \in \mathcal{H}_C^d$, as the canonical maps $\exp_x : T_x\mathcal{H}_C^d \to \mathcal{H}_C^d$ and $\log_x : \mathcal{H}_C^d \to T_x\mathcal{H}_C^d$ are known, and it is possible to reduce the learning of a representation $\mathbf{h}_v^{(l)} \in \mathcal{H}_C^d$ to the usual Euclidean operations on the tangent space $T_x\mathcal{H}_C^d$.

The quaternion set $\mathbb{H}$ is a four-dimensional vector space over the real numbers, consisting of one real and three imaginary parts, which can be thought of as a generalization of the complex numbers – therefore the denomination of hypercomplex number system. The vector space structure over $\mathbb{R}$ ensures that the product $x \otimes_{\mathbb{H}} y$ and sum $x \oplus_{\mathbb{H}} y$ operations above are well-defined and decomposable in terms of operation in $\mathbb{R}$. We follow [20] and choose $x \otimes_{\mathbb{H}} y$ to be the Hamilton product [16], which introduces weight sharing between the four components of the quaternion representation.

Hyperbolic and Quaternion models. The architectures of the models we developed follow the basic structure of the multitask GCN of [9], namely, a stack of graph convolutional layers followed

by fully connected layers before the final 10-dimensional output layer. This allows us to draw more precise conclusions pertaining the effects of geometric/algebraic features within the multitask setting. Next, we briefly illustrate some model-specific attributes and implementations.

Our multitask Hyperbolic GCN (HGCN) is based on [13], to which we refer for more details concerning the definition of hyperbolic feature transformation, neighbourhood aggregation and the readout function. One novelty of our implementation is hyperbolic batch normalization: we apply batch normalization of the features on tangent space before reprojection onto hyperbolic space. This improves numerical stability of the network, allowing for the use of a higher learning rate and fewer steps necessary to train to similar accuracy.

Each hyperbolic convolutional layer maps the embedding to a separate hyperboloid manifold, whose curvature is included as a learnable parameter in model training, allowing the model to find an optimal geometry for the data provided. This proves to be beneficial to model performance and model selection. In experimentation with deeper networks we consistently observed that the manifold curvature in early layers decreases to very low values, where the hyperboloid reduces to an almost-Euclidean geometry, while in deeper layers the model learns higher values of the curvature, seemingly exploiting the underlying hyperbolic features. This observation prompted us to explore a hybrid HGCN (HHGCN), where the first layer is replaced with a Euclidean-based graph convolutional layer. Both the fully hyperbolic and the hybrid model consist of 2 convolutional layers of size $[128, 128]$, with a dropout on the second convolution. These are followed by two fully connected layers of size $[128, 64]$. We use the AdamW optimizer with an initial learning rate of 0.001, the ReLU activation function and $\log\cosh$ loss function. The HGCN and HHGCN models consist of 69,197 and 272,076 trainable parameters, respectively.

Our implementation of the multitask Quaternion GNN (QGNN) introduces a few novelties with respect to the QGNN introduced by [20], to which we refer for the definition of the various basic layer-wise operations.

First, we define a new approach to batch normalization in quaternion space. Let $\mathbf{h} = [\mathbf{h}_r, \mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_k] \in \mathbb{H}^d$, where $\mathbf{h}_{r,i,j,k} \in \mathbb{R}^d$ are the real and three complex components. In our implementation we define quaternion batch normalization to be applied separately in $\mathbb{R}^d$ to each of the four components $\mathbf{h}_{r,i,j,k}$. This approach improves tremendously the learning stability and the overall performance of the model, in particular if the normalization occurs before the non-linearity transformation.

We also implement a recent node embedding strategy in quaternion space, where the map between the node features and the initial quaternion representation is learned during training. Namely, we apply the strategy of [21] and we introduce an embedding layer $\mathrm{Emb} : \mathbb{Z}^4 \to \mathbb{H}^d$. We denote this architecture Embedding QGNN (EQGNN). This has two potential advantages: first, we avoid initializing the representation component $\mathbf{h}_r = \mathbf{h}_i = \mathbf{h}_j = \mathbf{h}_k$ as suggested in [20], reducing the known detrimental effect that symmetries in weight space have on learning; second, we allow the network to learn the optimal embedding for solving the task at hand, increasing the model capacity.

Finally, we implement a simplified version of node soft-attention. We follow the proposal of [22] and assign attention scores to each hidden node embedding. The attention scores are obtained by applying a sigmoid activation function on the affine-transformed concatenation of node-embeddings. Each concatenated node-embedding is then multiplied element-wise with the attention scores before the global sum-readout function is applied. Our quaternion models have 3 quaternion layers of size $[32, 64, 128]$ with increasing amount of dropout followed by two fully connected layers of size $[256, 64]$. We use ReLU as activation function and $\log\cosh$ as our loss function. The initial learning rate of the AdamW optimizer is 0.001, which we decrease during training. The EQGNN and QGNN models consist of 380,558 and 467,118 trainable parameters, respectively.

Dataset. The datasets used in this work were assembled in [9] from the in-house Bayer database. It contains 10 different endpoints of biological and physicochemical assays including human serum albumin binding, membrane affinity, various solubility measures, lipophilicity (logD), and melting point. The in total 474,546 compounds are split into train/test sets based on molecular similarity clusters in order to increase the task difficulty, simulating model performance on novel compound classes. For further details on data preprocessing we refer the reader to [9].

Table 1: Comparison of performance (R^2 = coefficient of determination, ρ = Spearman's rank correlation coefficient) between the models with leave-cluster-out test set. In bold are reported the models with the highest R^2 and when equal, with the highest ρ.

		GCN		HGCN		HHGCN		EQGNN		QGNN	
Task[†]	Size (Train/Test)	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ	R^2	ρ
LOD	67,961 / 7,861	0.87	0.96	0.90	0.95	**0.91**	**0.96**	**0.91**	**0.96**	**0.91**	**0.96**
LOA	209,475 / 26,780	0.87	0.95	0.87	0.94	0.90	0.95	**0.91**	**0.96**	**0.91**	**0.96**
LOM	55,417 / 8,495	0.68	0.82	0.68	0.81	0.69	0.82	**0.71**	**0.84**	**0.71**	**0.84**
LOH	52,550 / 8,242	0.48	0.78	0.57	0.78	0.59	0.79	**0.60**	**0.79**	**0.60**	**0.79**
LMP	80,947 / 9,619	0.54	0.76	0.56	0.75	0.58	0.76	0.58	0.76	**0.59**	**0.77**
LOO	34,885 / 3,956	0.64	0.81	0.63	0.80	0.65	0.81	0.67	0.83	**0.68**	**0.83**
LOP	2,144 / 190	0.24	0.59	0.33	0.62	0.39	0.67	**0.45**	**0.70**	**0.45**	**0.70**
LON	75,385 / 12,915	0.65	0.80	0.65	0.80	0.67	0.81	0.66	0.81	**0.67**	**0.82**
LOX	6,655 / 737	0.63	0.80	0.67	0.81	0.68	0.81	**0.68**	**0.82**	**0.68**	**0.82**
LOQ	43,650 / 6,366	0.65	0.82	0.65	0.81	0.66	0.82	0.66	0.82	**0.66**	**0.83**

Molecular graphs were generated with the featurizer used in [9]: each molecule is one-hot encoded into a graph, with 75 features per atom describing atom type, charge, number of radical electrons, aromaticity, hybridization, valence, degree, and number of H-bonds. For the EQGNN model, the one-hot encoded node representation is further transformed into an integer-based one. We remark that our proposed models do not include edge attributes between connected atoms.

3 Experiments

We evaluate the different network designs and geometries we introduced in the previous section on a large ADMET dataset. In Table 1 we list the performance of our models in a held-out leave-cluster-out test set. We compare our results with the GCN model from [9], which is currently the best performing model on this dataset. We do take care of evaluating our models on exactly under the same conditions, that is, we do not perform any extensive hyperparameter or model selection. Our models were trained for a fixed number of 50 epochs, and no early-stopping or other validation technique has been used. Compared to the model in [9] with 613,642 trainable parameters, our 4 models require less trainable parameters.

We observe that the quaternion-based networks QGNN and EQGNN perform the best among our models. These outperforms the Euclidean-based GCN on several tasks and yield a higher average R^2 over all tasks ($+0.06$ for both models). Perhaps more interesting, (E)QGNN perform particularly well on very small tasks: on LOP (2144 compounds) and LOX (6655 compounds), the two smallest tasks in the dataset, the model shows an average performance increase of $+0.13$ in R^2 and $+0.06$ in Spearman's ρ. This result seems to support our hypothesis that quaternion space, endowed with the Hamilton product, is especially suited for multitask learning, as it prevents overfitting over the bigger tasks.

We observe that our versions of Hyperbolic GCN also show strong performance across all tasks, with the hybrid model performing better or equal to the GCN in all tasks. The HGCN and HHGCN models report an overall average R^2 increase of $+0.03$ and $+0.05$, respectively.

4 Conclusions

In this work we implemented two recent geometric approaches to Deep Learning in the context of multitask learning of ADMET properties. We showed that all our models show strong performance across all tasks. In particular, our quaternion models (E)QGNN outperform the GCN network of [9] in all tasks. The large performance gain on the smallest task supports our hypothesis that quaternion space is particularly suitable in the context of multitask learning and inhomogeneous datasets. Despite

[†]LOD=LogD in neutral pH, LOA=LogD in acidic pH, LOM = membrane affinity, LOH = human serum albumin binding, LMP = melting point, LOO = DMSO solubility, LON = nephelometric solubility, LOQ = solubility without assay annotation, LOP = powder solubility, LOX = DMSO not fully dissolved solubility.

showing a slightly inferior performance than the quaternion models, our hyperbolic models (H)HGCN do however perform better than the GCN model. The discrepancy with respect to the hypercomplex case might arise from the extra challenges introduced by training in hyperbolic space, due to the extra maps to and from the tangent space, yielding a less treatable loss objective. The graph size could also play a decisive role: the molecules in our dataset are relatively small, while it has been argued [15] that hyperbolic geometry is particularly suitable for large scale graph analysis. Also, the GCN model of [9] is based on the graph convolution method introduced in [1], while our hyperbolic model is adapted on the standard graph convolution [23, 24]. It would be interesting to extend the approach of [1] to the hyperbolic world and quantify the difference in performance. We plan to return to this in future work.

Acknowledgments and Disclosure of Funding

MB, FM & DAC acknowledge funding from the Bayer AG Life Science Collaboration ("Explainable AI","DeepMinDS"). TL acknowledges funding from the Bayer AG PhD scholarship.

References

[1] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Advances in Neural Information Processing Systems 28* (C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds.), pp. 2224–2232, Curran Associates, Inc., 2015.

[2] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," vol. 70 of *Proceedings of Machine Learning Research*, (International Convention Centre, Sydney, Australia), pp. 1263–1272, PMLR, 06–11 Aug 2017.

[3] Z. Wu, B. Ramsundar, E. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande, "Moleculenet: a benchmark for molecular machine learning," *Chem. Sci.*, vol. 9, pp. 513–530, 2018.

[4] K. Yang, K. Swanson, W. Jin, C. Coley, P. Eiden, H. Gao, A. Guzman-Perez, T. Hopper, B. Kelley, M. Mathea, A. Palmer, V. Settels, T. Jaakkola, K. Jensen, and R. Barzilay, "Analyzing learned molecular representations for property prediction," *Journal of Chemical Information and Modeling*, vol. 59, no. 8, pp. 3370–3388, 2019. PMID: 31361484.

[5] D. Rogers and M. Hahn, "Extended-connectivity fingerprints," *Journal of Chemical Information and Modeling*, vol. 50, no. 5, pp. 742–754, 2010. PMID: 20426451.

[6] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, p. 41–75, July 1997.

[7] S. Ruder, "An overview of multi-task learning in deep neural networks," *CoRR*, vol. abs/1706.05098, 2017.

[8] S. Kearnes, B. Goldman, and V. Pande, "Modeling industrial admet data with multitask networks," arXiv:stat.ML/1606.08793, 2016.

[9] F. Montanari, L. Kuhnke, A. Ter Laak, and D.-A. Clevert, "Modeling physico-chemical admet endpoints with multitask graph convolutional networks," *Molecules*, vol. 25, no. 1, 2020.

[10] F. Capela, V. Nouchi, R. V. Deursen, I. V. Tetko, and G. Godin, "Multitask learning on graph neural networks applied to molecular property predictions," arXiv:cs.LG/1910.13124, 2019.

[11] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5425–5434, 2017.

[12] O. Ganea, G. Becigneul, and T. Hofmann, "Hyperbolic neural networks," in *Advances in Neural Information Processing Systems 31* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), pp. 5345–5355, Curran Associates, Inc., 2018.

[13] I. Chami, Z. Ying, C. Ré, and J. Leskovec, "Hyperbolic graph convolutional neural networks," in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 4868–4879, Curran Associates, Inc., 2019.

[14] Q. Liu, M. Nickel, and D. Kiela, "Hyperbolic graph neural networks," in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8230–8241, Curran Associates, Inc., 2019.

[15] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Advances in Neural Information Processing Systems 30* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 6338–6347, Curran Associates, Inc., 2017.

[16] C. J. Gaudet and A. S. Maida, "Deep quaternion networks," in *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2018.

[17] X. Zhu, Y. Xu, H. Xu, and C. Chen, "Quaternion convolutional neural networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[18] T. Parcollet, M. Ravanelli, M. Morchid, G. Linarès, C. Trabelsi, R. D. Mori, and Y. Bengio, "Quaternion recurrent neural networks," in *International Conference on Learning Representations*, 2019.

[19] T. Parcollet, M. Morchid, and G. Linarès, "Quaternion convolutional neural networks for heterogeneous image processing," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8514–8518, 2019.

[20] D. Q. Nguyen, T. D. Nguyen, and D. Phung, "Quaternion graph neural networks," arXiv:cs.LG/2008.05089, 2020.

[21] W. Hu*, B. Liu*, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec, "Strategies for pre-training graph neural networks," in *International Conference on Learning Representations*, 2020.

[22] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," vol. 80 of *Proceedings of Machine Learning Research*, (Stockholmsmässan, Stockholm Sweden), pp. 5453–5462, PMLR, 10–15 Jul 2018.

[23] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, 2017.

[24] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems 30* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 1024–1034, Curran Associates, Inc., 2017.