Tree Covers: An Alternative to Metric Embeddings

Roshni Sahoo, Ines Chami & Christopher Ré
Stanford University
`{rsahoo,chami,chrismre}@cs.stanford.edu`

Abstract

We study the problem of finding distance-preserving graph representations. Most previous approaches focus on learning continuous embeddings in metric spaces such as Euclidean or hyperbolic spaces. Based on the observation that embedding into a metric space is not necessary to produce faithful representations, we explore a new conceptual approach to represent graphs using a collection of trees, namely a *tree cover*. We show that with the same amount of storage, covers achieve lower distortion than learned metric embeddings. While the distance induced by covers is not a metric, we find that tree covers still have the desirable properties of graph representations, including efficiency in query and construction time.

1 Introduction

The study of distance-preserving graph embeddings [5] has attracted interest in the past decades due to the widespread presence of graphs in real-world applications such as biology [23], chemistry [11], recommender systems [24], and knowledge bases [4, 6]. Embeddings into simpler geometric spaces can allow us to transform complicated problems on the original graph (e.g. nearest neighbor search) into easier ones in the embedding space. Desirable properties for these representations include:

1. *faithfulness*, i.e. how well they preserve distances from the original metric [12, 20],
2. *efficiency*, i.e. how fast they can be built and queried.

The dominant approach to finding such representations is embedding into low-dimensional Euclidean spaces. However, these embeddings may not achieve low distortion, meaning that the embedding distances may not closely approximate the graph distances. Embeddings into non-Euclidean metric spaces yield low-distortion representations when the geometry of the space matches the structure of the data [12, 15]. For instance, trees can be represented with arbitrarily low distortion in hyperbolic space [20, 21], while Bourgain's theorem shows that this is not possible in Euclidean space [14].

Selecting the right metric space to embed an arbitrary graph is a challenging problem. Our work builds on the observation that in many cases, the metric structure may not be essential for obtaining *faithful* embeddings. Any low-distortion embedding (whether it lies in a metric space or not) has the desirable property that it preserves the structure of the data *faithfully*.

This motivates us to consider a new conceptual approach of representing graphs using *tree covers*, which have generated interest in theoretical computer science because they can approximate the original graph distances more faithfully than metric embeddings. A tree cover of a graph is a collection of trees where every pair of vertices has a low-distortion path in one of the trees. The distance function induced by tree covers is not a metric (taking the minimum over all the distances in the cover components means that the triangle inequality is not satisfied), but they can still achieve low distortion. For example, the metric of the n-point cycle incurs $\Omega(n)$ distortion when embedding into a tree metric [18], but a tree cover on just 2 trees can preserve the distances exactly. Although polynomial-time algorithms for constructing tree covers have been studied, they are not practical for real-world graphs.

In this paper, we propose a greedy tree cover algorithm to compute *faithful* and *efficient* graph representations.

1. Given an unweighted graph, our algorithm sequentially adds shortest-path trees to the cover, with the goal of ensuring that there exists a short path between each node and a root of a tree in the cover.
2. Our proposed algorithm constructs a k-tree cover in $O(kn^2)$-time and can answer queries *efficiently* by leveraging the simple structure of trees.

We validate our method on real-world graphs and find that tree covers achieve lower distortion than metric embedding approaches with the same amount of storage. Finally, we study the performance of covers compared to learned metric embeddings, and find that covers yield at least a 1750x speedup in construction-time on these real datasets. Some of this efficiency is traded off at query-time.

2 Related Work

Embeddings into low-dimensional spaces. Continuous representations are efficient in terms of query-time and storage; a d-dimensional embedding of a graph on n nodes permits $O(d)$-time distance queries and requires nd parameters. When the structure of the graph matches the geometry of the embedding space, these embeddings can achieve low distortion. Hyperbolic geometry is known to embed trees and hierarchical data faithfully [7, 15, 20, 21], and analogous connections exist between spherical geometry and cycles [20], as well as product spaces and graphs with a mix of treelike and cyclical structures [12]. Nevertheless, determining the appropriate metric space to embed an arbitrary graph is challenging.

Embeddings into tree metrics. Many combinatorial problems on graphs are easier to solve if the graph is a tree, motivating many works on embedding into tree metrics [1, 10]. Trees require low storage and allow for constant-time queries with the help of short distance-labeling schemes [9]. However, the distortion achieved by a tree embedding is tied to the δ-hyperbolicity of the original metric, or how far the original metric is from a tree metric [1, 22]. Embedding into a tree metric may not yield a faithful representation if the original metric has high δ-hyperbolicity.

Low-distortion tree covers. Previous works on tree covers derive bounds on the number on trees needed to achieve a particular level of distortion for general metrics, as well as planar and doubling metrics [3, 8, 13]. For a general metric on n points, a tree cover requires $\Omega(n^{1/\alpha})$ trees to achieve distortion α [3]. Although not as much is known about the distortion that can be achieved when the number of trees in the cover is small, there is a deterministic algorithm for constructing a Ramsey tree cover of size k which achieves distortion $O(n^{1/k} \cdot (\log n)^{1-1/k})$ for a general n-point metric [3]. A Ramsey tree cover has the stronger property that for every node there exists a tree in the cover in which the node has low-distortion paths to all other nodes. To the best of our knowledge, we are the first to evaluate the quality and performance of these tree cover algorithms in a practical setting.

3 Model

3.1 Problem setup

Previous works treat learning graph representations as a metric embedding problem. We describe our relaxation of the metric embedding setting and define *average distortion*, a standard measure for assessing the faithfulness of the embedding.

Problem definition. Given a connected, unweighted graph $G = (V, E)$ on n nodes and its shortest path metric $d_G(\cdot, \cdot)$, we aim to find an embedding $f : G \to X$, where X has a symmetric function $d_X(\cdot, \cdot) : X \times X \to \mathbb{R}_{\geq 0}$ and $d_X(x, y) = 0$ iff $x = y$, such that the average distortion:

$$d_{\text{ave}} = \frac{1}{\binom{n}{2}} \sum_{u,v \in V : u \neq v} \frac{|d_X(f(u), f(v)) - d_G(u, v)|}{d_G(u, v)}$$

is minimized. Note that standard metric embedding adds the requirement that (X, d_X) is a metric space, i.e. $d_X(\cdot, \cdot)$ satisfies the triangle inequality. We relax this constraint here and consider tree covers to obtain faithful graph representations.

		Diseasome	Yeast	CS PhD	GridMouse	GridWorm
		$\lvert V\rvert = 516$ $\lvert E\rvert = 1188$	$\lvert V\rvert = 1458$ $\lvert E\rvert = 1948$	$\lvert V\rvert = 1025$ $\lvert E\rvert = 1043$	$\lvert V\rvert = 791$ $\lvert E\rvert = 1098$	$\lvert V\rvert = 3343$ $\lvert E\rvert = 6437$
Metric	$\mathbb{E}^{10}$	0.0562	0.0795	0.0558	0.0601	0.0970
	$\mathbb{H}^{10}$	0.0402	0.0742	0.0508	0.0531	0.0795
	$\mathbb{S}^{10}$	0.0602	0.0811	0.0581	0.0634	0.1083
	$(\mathbb{H}^2)^5$	0.0344	0.0706	0.0375	0.0445	0.0792
	$(\mathbb{H}^5)^2$	0.0354	0.0713	0.0345	0.0466	0.0773
	$(\mathbb{S}^2)^5$	0.0583	0.0821	0.0565	0.0630	0.1096
	$(\mathbb{S}^5)^2$	0.0595	0.0798	0.0588	0.0630	0.0998
Covers	BFS (5 Trials)	0.0280	0.0529	0.0033	0.0206	0.0660
	MST (5 Trials)	0.3617	0.3026	0.0058	0.1364	0.5451
	Ramsey	0.0298	-	-	0.0111	-
	Ours	**0.0213**	**0.0208**	**0.0001**	**0.0056**	**0.0347**

Table 1: Average distortion for different methods. The dash indicates the experiment took more than 5 hours to run.

Spanning tree covers. Given a graph $G = (V, E)$ and its shortest path metric d_G, an (α, k)-spanning tree cover is a collection of k *spanning trees* of G, $\mathcal{T} = \{T_1, T_2 \dots T_k\}$ such that: for every pair of vertices $u, v \in V$, there exists a path between u, v in one of the trees that achieves at most α-distortion. We define the distance between two vertices $u, v \in V$ in the tree cover as follows:

$$d_{\mathcal{T}}(u, v) = \min_{i \in [k]} d_{T_i}(u, v).$$

We note that the cover distance dominates the graph metric, i.e. $d_G(u, v) \le d_{\mathcal{T}}(u, v)\ \forall u, v \in V$.

3.2 Cover Algorithm

We build a tree cover by selecting k shortest-path trees of the graph $G = (V, E)$. A shortest-path tree rooted at a node r is a spanning tree T of G, such that $d_T(r, v) = d_G(r, v)$ for any $v \in V$. In an unweighted graph, the shortest-path tree rooted at r can be obtained via breadth-first search (BFS) from r.

The problem of selecting the optimal k shortest-path trees such that the distortion is minimized is an NP-hard problem [17], so we propose GreedyTreeCover, a greedy algorithm for selecting the trees. Its inputs include the graph G, the shortest-path metric d_G, and the number of trees in the cover k. We aim to select a set of root nodes $\mathcal{R} = \{r_1, r_2, \dots r_k\}$ of the shortest-path trees $\{T_1, T_2, \dots T_k\}$ so that each vertex $v \in V$ is close to one of the cover roots.

We maintain a set of root nodes $\mathcal{R}_t$ of size t chosen thus far and an array A maintaining each node's current depth in the cover, so $A_{\mathcal{R}_t}[v] = \min_{r \in \mathcal{R}_t} d_G(r, v)$. We select r_{t+1} by finding the node that decreases in the cover depths most significantly, relative to the previous depths, which takes $O(n^2)$ time. Finally, we set $\mathcal{R}_{t+1} = \mathcal{R}_t \cup \{r_{t+1}\}$ and update the node depths in the cover. Pseudocode for this algorithm is provided in the appendix (Appendix B). The intuition for this algorithm is that the distance between two nodes u, v in a tree T rooted at r can be written as $d_T(u, v) = d_T(r, u) + d_T(r, v) - 2d_T(r, u \vee v)$, where $u \vee v$ is the lowest common ancestor of u and v in T. If $r = u \vee v$, then shrinking $d_T(r, v)$ reduces $d_T(u, v)$.

GreedyTreeCover produces a k-tree cover of the graph in $O(kn^2)$ time. Storing the tree cover requires $O(nk)$ space, as we store the $n - 1$ edges of each tree. Performing a distance query requires computing a distance in each tree, taking $O(nk)$-time overall. There are many strategies for reducing the query-time, such as storing distance labels for each node so that distances can be queried in constant-time [9].

	Diseasome		Yeast		CS PhD		GridMouse		GridWorm	
	CT	QT	CT	QT	CT	QT	CT	QT	CT	QT
Metric	$3.5e^5$	0.846	$1.6e^6$	0.896	$5.6e^5$	0.36	$3.4e^5$	0.359	$3.6e^6$	0.851
BFS	53.9	0.313	195	0.428	120	0.97	91.89	0.564	489	0.510
MST	132	0.727	302.8	1.67	183	1.42	159	1.082	997	3.907
Ramsey	$3.9e^6$	0.385	$>1.8e^7$	-	$>1.8e^7$	-	$1.7e^7$	0.815	$>1.8e^7$	-
Ours	140	0.292	397	0.384	255	1.00	195	0.584	1045	0.379

Table 2: Performance analysis of the construction time (CT) and query time (QT) in milliseconds.

4 Experiments

4.1 Experimental Setup

Datasets. We use 5 real-world graph datasets: CS PhD, a treelike graph of Ph.D. advisor-advisee relationships [16] and four biological network datasets which vary in their density from [19].

Baselines. We compare GreedyTreeCover to the following baselines. First, we compare to learned metric embeddings, such as Euclidean ($\mathbb{E}^d$), hyperbolic ($\mathbb{H}^d$), spherical ($\mathbb{S}^d$), and product space embeddings. We train these baselines with the distortion-based loss function from [12]. In addition, we compare to three discrete tree cover algorithms: BFS, MST, and Ramsey. BFS forms a tree cover by randomly selecting shortest-path trees. MST forms a tree cover by repeating the follow procedure: assign random weights sampled from a uniform distribution to the edges of the graph, run Kruskal's minimum spanning tree algorithm, and add the resulting (unweighted) tree to the cover. Lastly, Ramsey denotes a discrete spanning tree cover algorithm which has theoretical guarantees [3]. The cover is generated by running the petal decomposition algorithm [2] to create low-distortion spanning trees until the Ramsey criteria is met. If the criteria is met prior to obtaining k trees, we run petal decomposition from randomly selected initial nodes until k trees are generated.

4.2 Distortion Results

For fair comparison, we fix the amount of storage across all methods. We consider 10-dimensional continuous embeddings and tree covers on 10 trees. The results on BFS and MST are averaged over 5 trials. We observe that our greedy algorithm obtains the lowest distortion across all datasets (Table 1).

4.3 Performance Analysis

We compare the construction-time (CT) and query-times (QT) of the metric embedding method that achieves lowest distortion to those of discrete tree covers. The metric embedding methods that perform best on these datasets are the product space embeddings into $(\mathbb{H}^2)^5$ and $(\mathbb{H}^5)^2$. CT results are averaged over 5 trials and QT results are the estimated time per query over 2000 queries. We observe that our method achieves at least a 1750x speedup in CT relative to the metric embedding methods (Table 2). The CT of BFS and MST are even faster than our method but typically achieve worse distortion. Ramsey takes over 5 hours to run on Yeast, CS PhD, and GridWorm. GreedyTreeCover trades off its efficiency in CT with slower queries. Our method's QT typically scales with the number of nodes in the graph. In contrast, we find that the QT of the metric embeddings depends on the number of components in the embedding space because a separate distance computation is required for each component.

5 Conclusion

We introduced the GreedyTreeCover algorithm and observed that discrete tree covers can achieve lower distortion than metric embeddings while using the same amount of storage. We believe that relaxing the requirement to maintain the metric structure in graph representations will open a new avenue to more efficient and faithful representations than classical embedding methods.

Acknowledgments and Disclosure of Funding

We gratefully acknowledge the support of DARPA under Nos. FA86501827865 (SDH) and FA86501827882 (ASED); NIH under No. U54EB020405 (Mobilize), NSF under Nos. CCF1763315 (Beyond Sparsity), CCF1563078 (Volume to Velocity), and 1937301 (RTML); ONR under No. N000141712266 (Unifying Weak Supervision); the Moore Foundation, NXP, Xilinx, LETI-CEA, Intel, IBM, Microsoft, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, the Okawa Foundation, American Family Insurance, Google Cloud, Swiss Re, the HAI-AWS Cloud Credits for Research program, NSF GRFP under grant number DGE-1656518, and members of the Stanford DAWN project: Teradata, Facebook, Google, Ant Financial, NEC, VMWare, and Infosys. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of DARPA, NIH, ONR, or the U.S. Government.

References

[1] Ittai Abraham, Mahesh Balakrishnan, Fabian Kuhn, Dahlia Malkhi, Venugopalan Ramasubramanian, and Kunal Talwar. Reconstructing approximate tree metrics. pages 43–52, 08 2007.

[2] Ittai Abraham and Ofer Neiman. Using petal-decompositions to build a low stretch spanning tree. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '12, page 395–406, New York, NY, USA, 2012. Association for Computing Machinery.

[3] Yair Bartal, Nova Fandina, and Ofer Neiman. Covering metric spaces by few trees. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[5] Ines Chami, Sami Abu-El-Haija, Bryan Perozzi, Christopher Ré, and Kevin Murphy. Machine learning on graphs: A model and comprehensive taxonomy. *arXiv preprint arXiv:2005.03675*, 2020.

[6] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6901–6914, Online, July 2020. Association for Computational Linguistics.

[7] Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. Hyperbolic graph convolutional neural networks. In *Advances in neural information processing systems*, pages 4868–4879, 2019.

[8] T.-H. Hubert Chan, Anupam Gupta, Bruce M. Maggs, and Shuheng Zhou. On hierarchical routing in doubling metrics, Jun 2018.

[9] Ofer Freedman, Pawel Gawrychowski, Patrick Nicholson, and Oren Weimann. Optimal distance labeling schemes for trees. pages 185–194, 07 2017.

[10] Jie Gao and Dengpan Zhou. Resilient and low stretch routing through embedding into tree metrics. pages 438–450, 08 2011.

[11] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*, 2017.

[12] Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. Learning mixed-curvature representations in product spaces. In *International Conference on Learning Representations*, 2018.

[13] Anupam Gupta, Amit Kumar, and Rajeev Rastogi. Traveling with a pez dispenser (or, routing issues in mpls). *SIAM J. Comput.*, 34(2):453–474, February 2005.

[14] Nathan Linial, E London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–246, 03 2001.

[15] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6338–6347. Curran Associates, Inc., 2017.

[16] Wouter De Nooy, Andrej Mrvar, and Vladimir Batagelj. *Exploratory Social Network Analysis with Pajek*. Cambridge University Press, USA, 2011.

[17] Michalis Potamias, Francesco Bonchi, Carlos Castillo, and Aristides Gionis. Fast shortest path distance estimation in large networks. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, page 867–876, New York, NY, USA, 2009. Association for Computing Machinery.

[18] Yuri Rabinovich and R. Raz. Lower bounds on the distortion of embedding finite metric spaces in graphs. *Discrete and Computational Geometry*, 19:79–94, 01 1998.

[19] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

[20] Frederic Sala, Chris De Sa, Albert Gu, and Christopher Ré. Representation tradeoffs for hyperbolic embeddings. In *International Conference on Machine Learning*, pages 4460–4469, 2018.

[21] Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *International Symposium on Graph Drawing*, pages 355–366. Springer, 2011.

[22] Rishi Sonthalia and Anna C. Gilbert. Tree! i am no tree! i am a low dimensional hyperbolic embedding, 2020.

[23] Chris Stark, Bobby-Joe Breitkreutz, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, and Mike Tyers. Biogrid: a general repository for interaction datasets. *Nucleic acids research*, 34(suppl_1):D535–D539, 2006.

[24] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, 52(1), February 2019.

A Definitions

Dominating. For two metric spaces G and X defined on the same vertex set V, X *dominates* G if $d_X(u,v) \geq d_G(u,v)$ for all $u,v \in V$.

Multiplicative Distortion. Given an embedding f from a metric space (G, d_G) to a metric space (X, d_X) which dominates G, we say that f obtains multiplicative distortion α if, for all $u,v \in G$, it holds that

$$d_X(f(u), f(v)) \leq \alpha \cdot d_G(u,v).$$

B Cover Algorithm

We provide pseudocode for the tree cover algorithm. Suppose that ShortestPathTree(G, r) is a helper function that returns the shortest path rooted at node r in a graph G.

Algorithm 1: GreedyTreeCover(G, k, d_G)

Input: An unweighted, connected graph G, the number of trees in the cover k, an $n \times n$ matrix representing the shortest path metric d_G.
Output: A set of k shortest path trees

```
Select tree with lowest average node depth and add to cover.
r_1 ← argmin_{i∈[n]} Σ_j d_G[i][j]
Initialize tree cover roots.
R ← {r_1}
Initialize cover depths.
A ← d_G[r_1]
for 1 ≤ i < k do
    The next tree added to the cover yields the most significant reduction in cover depths, relative
     to the previous depths.
    M ← 0
    r_i ← None
    for 0 ≤ v < n and v ∉ R do
        B ← d_G[v]
        m ← Σ_{j∈V\R} max(A[j] − B[j], 0)/A[j]
        if m > M then
            M ← m
            r_i ← v
    Add r_i to the cover and update cover depths.
    R ← R ∪ {r_i}
    A ← [min(A[j], d_G[r_i][j]) for j ∈ V].
T ← {ShortestPathTree(G, r) for r ∈ R}
return T
```

C Additional Distortion Results

We evaluate on the same datasets described in Section 4.1. We compare a tree cover on d trees generated via GreedyTreeCover to learned metric embeddings into Euclidean $(\mathbb{E}^d)$ and hyperbolic $(\mathbb{H}^d)$ spaces. For fair comparison, the amount of storage used across all methods is fixed. We consider d-dimensional metric embeddings and tree covers on d trees where $d \in [2, 4, 8, 16, 32]$. In Figure 1, We observe that our method achieves lower distortion at these dimensions. The tree cover approaches continue to improve as the dimension is increased, but the metric embeddings into Euclidean and hyperbolic spaces appear to plateau earlier.

D Experiment Details

For training the metric embeddings in Section 4.1, we use the setup provided in [12], taken from `https://github.com/HazyResearch/hyperbolics`. The optimization framework was implemented in PyTorch. The distortion-based loss function was optimized with SGD using minibatches of 65536 edges, and we trained

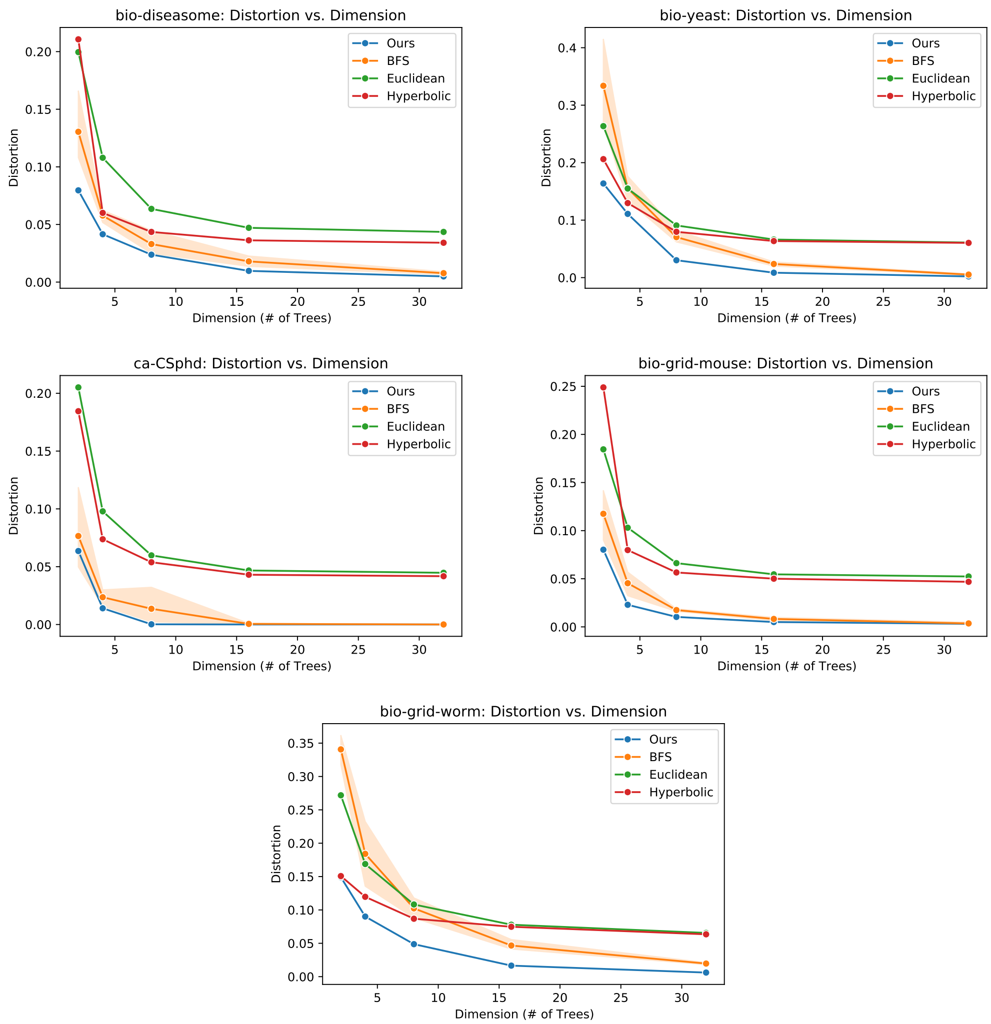

Figure 1: The average distortion across all methods improves as dimensionality increases. Our method achieves lower distortion than the baselines across the evaluated dimensions.

for 5000 epochs. We specified the following commandline arguments: –resample-freq 5000, -g, –subsample 1024, and –riemann. The learning rate was chosen from a grid search among $\{10, 30, 100, 300, 1000\}$ for each method.