From Trees to Continuous Embeddings and Back: Hyperbolic Hierarchical Clustering

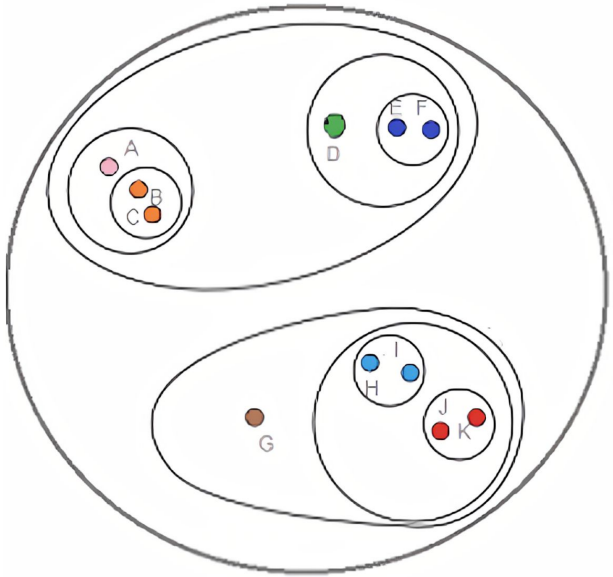
Ines Chami, Albert Gu, Vaggos Chatziafratis, Christopher Ré

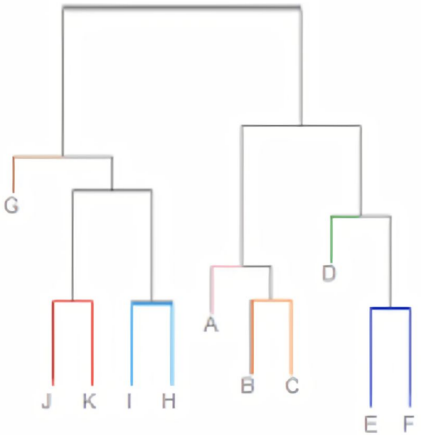
Continuous Embeddings

- A continuous embedding of a graph (for example, a tree) is a way of representing a graph with vectors in a continuous space.
- In this paper, trees are represented with continuous vectors where each leaf node in the tree is represented by a position.
- The positions of the leaf nodes determine the structure of the tree.
- For each continuous embedding their is a unique tree, but for each tree there are infinitely many embeddings.

Hierarchical Clustering

Given a set of data points and similarities between data points, cluster them using a binary tree where leaves represent data points and internal nodes represent hierarchical clustering of data points.





Dasgupta's cost

$$C_{\text{Dasgupta}}(T; w) = \sum_{ij} w_{ij} |\text{leaves}(T[\text{LCA(i, j)}])|$$

Where w_ij is the similarity weight between i and j, and

LCA(i, j) is the lowest common ancestor of i and j on the tree T.

(the sum is also technically a sum over sets {i, j})

Dasgupta’s Cost Example

Let W_CD = 1, W_DE = 2, W_ij = 0 in all other cases.

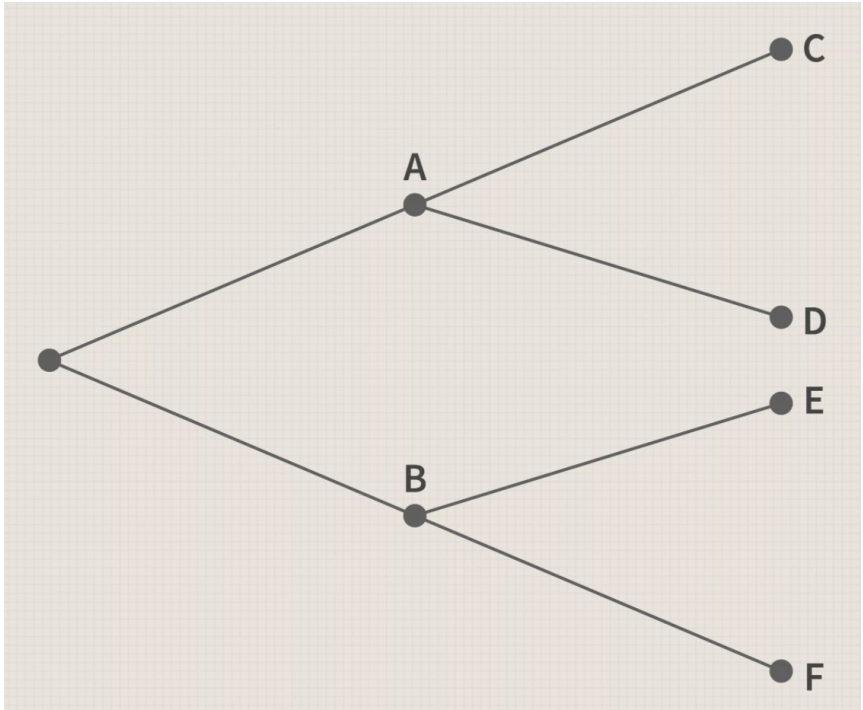
|leaves(LCA(C, D))| = |{C, D}| = 2

|leaves(LCA(D, E))| = |{C, D, E, F}| = 4

Dasgupta’s Cost = (W_CD * 2) + (W_DE * 4) = (1 * 2) + (2 * 4) = 10.

Is this optimal?

No, swap C and E.



Reformatting Dasgupta's cost

Define {i, j|k} as true if LCA(i, j) is a proper ancestor of LCA(i, j, k).

Leaves(T[LCA(i, j)]) is the set of k with {i, j| k} false, and thus

|Leaves(T[LCA(i, j)])| = $\sum_k (1 - 1_{\{i,j|k\}})$

$$C_{\text{Dasgupta}}(T; w) = \sum_{ijk} [w_{ij} + w_{ik} + w_{jk} - w_{ijk}(T; w)] + 2 \sum_{ij} w_{ij}$$

$$\text{where} \quad w_{ijk}(T; w) = w_{ij} \mathbb{1}[\{i, j|k\}] + w_{ik} \mathbb{1}[\{i, k|j\}] + w_{jk} \mathbb{1}[\{j, k|i\}],$$

(and where the sum is over sets {i, j, k} of size 3, rather than individually over i, k, and k, accounting for the factor of 3)

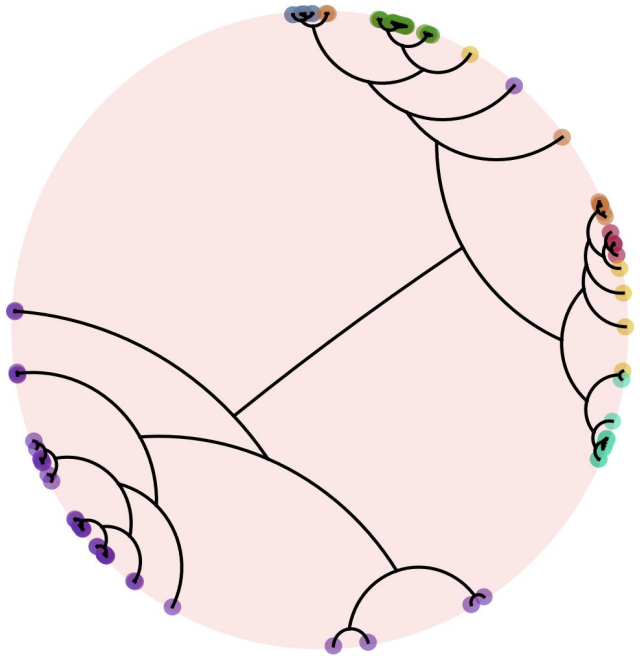
This reformatting helps with converting to a continuous version of Dasgupta's cost

Hyperbolic geometry

Curved geometry where distances increase for points away from the origin.

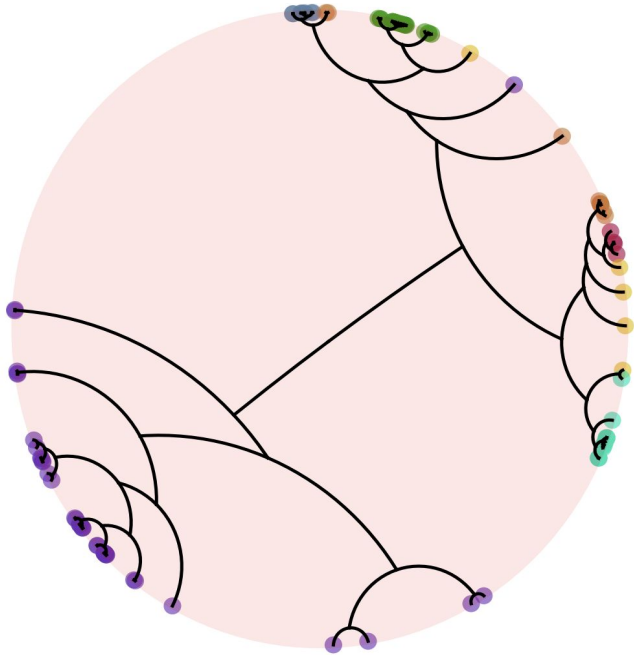
The shortest path between points (geodesics) is a curve towards the origin.

Connecting many points and midpoints of shortest paths gives tree like structures.



$$d(x,y) = \cosh^{-1}\left(1 + 2\frac{||x-y||_2^2}{(1-||x||_2^2)(1-||y||_2^2)}\right).$$

Hyperbolic Least Common Ancestor



Define LCA(x, y) for two points in hyperbolic space to the point on the shortest path connecting x to y which is closest to the origin.

This same concept applied to trees gives the lowest common ancestor.

Define d_0(z) as the distance from z to the origin.

If d_0 was applied to trees, {i, j|k} is true if and only if d_0(LCA(i, j)) < d_0(LCA(i, k)), and d_0(LCA(i, j)) < d_0(LCA(j, k)).

Thus, w_ij = $w_{ijk}(T; w) = w_{ij}\mathbb{1}[\{i, j|k\}] + w_{ik}\mathbb{1}[\{i, k|j\}] + w_{jk}\mathbb{1}[\{j, k|i\}]$

if and only if d_0(LCA(i, j)) < d_0(LCA(i, k)), and d_0(LCA(i, j)) < d_0(LCA(j, k))

Continuous version of Dasgupta's cost

Define $w_{\mathrm{HYPHC},ijk}(Z;w,\tau) = (w_{ij},w_{ik},w_{jk}) \cdot \sigma_\tau(d_0(LCA(z_i,z_j)),d_0(LCA(z_i,z_k)),d_0(LCA(z_j,z_k)))$

Where $\sigma_\tau(\cdot)$ is the scaled softmax function: $\sigma_\tau(\alpha)_i = e^{\alpha_i/\tau}/\sum_i e^{\alpha_j/\tau}$

As t approaches zero, $w_{\mathrm{HYPHC},ijk}(Z;w,\tau)$ approaches $w_{ijk}(T;w)$

$= w_{ij}\mathbb{1}[\{i,j|k\}] + w_{ik}\mathbb{1}[\{i,k|j\}] + w_{jk}\mathbb{1}[\{j,k|i\}]$

Define $C_{\mathrm{HYPHC}}(Z;w,\tau) = \sum_{ijk}(w_{ij}+w_{ik}+w_{jk}-w_{\mathrm{HYPHC},ijk}(Z;w,\tau)) + 2\sum_{ij} w_{ij}$

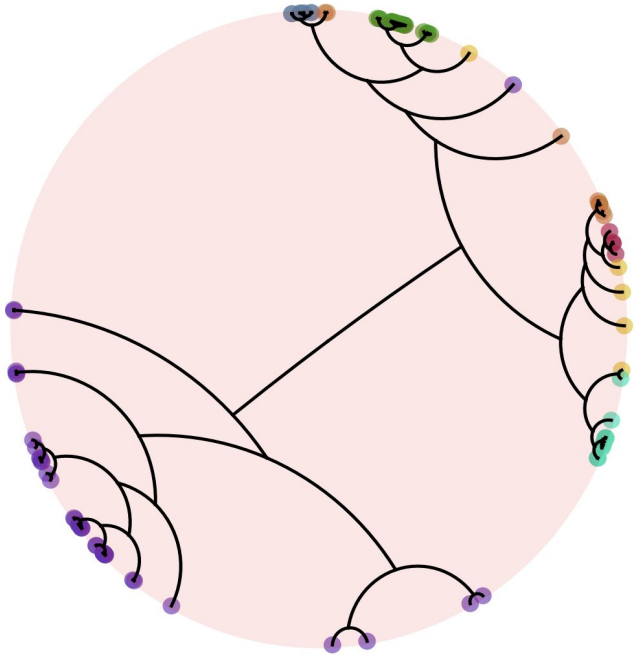
As t approaches zero, this approaches the Dasgupta's cost.

This modification is useful since it can be applied to positions z_i in hyperbolic space, and is a differentiable equation of the positions z_i.

Intuition on this continuous cost

To optimize the new cost function, one maximizes the closest distance to the origin in the shortest line connecting similar data points.

Similar data points are encouraged to be near each other and on small subtrees together.



Decoding continuous representations

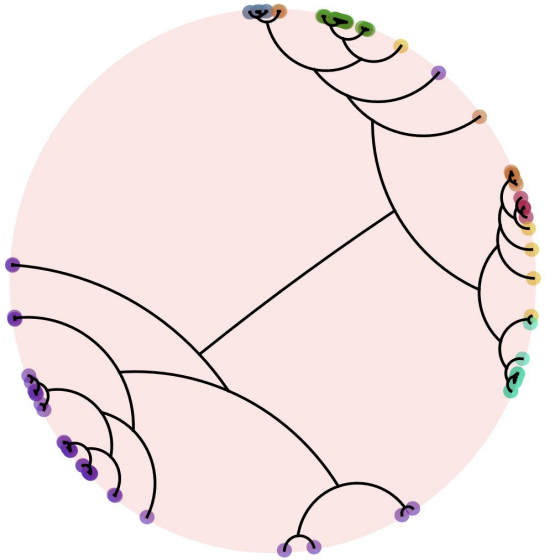
After optimizing the positions of the data points with the new continuous loss function, one must convert this back to a discrete tree.

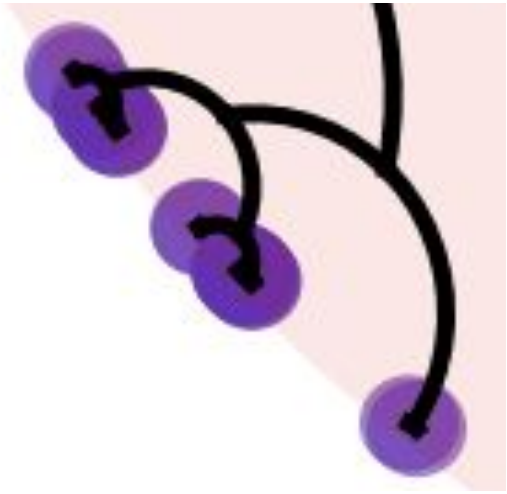
To do this, define the initial set of trees as trees of size 1 for every data point (leaf node).

Then, combine the two trees whose roots are the closest by adding their lowest common ancestor as a root.

Repeat this until there is only one large tree.

$$d(x,y) = \cosh^{-1}\left(1 + 2\frac{||x-y||_2^2}{(1-||x||_2^2)(1-||y||_2^2)}\right).$$





Results

This method (HypHC) is compared to common hyperbolic clustering methods. Specifically, the agglomerative clustering approaches “single”, “average” complete” and “Ward Linkage”.

HypHC outperforms all of these methods on Desgupta’s cost.

Note, this comparison is slightly biased by the fact that the agglomerative clustering approaches do not explicitly optimize a version of Desgupta’s cost.