Accelerating hyperbolic t-SNE

Quadtree generalization for the upper half-plane model

Diyan Dimitrov[1]

Supervisor(s): Martin Skrodzki[1]**, Elmar Eisemann**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Diyan Dimitrov
Final project course: CSE3000 Research Project
Thesis committee: Martin Skrodzki, Elmar Eisemann, Gosia Migut

Abstract

Dimensionality reduction is essential for analyzing high-dimensional datasets across various fields. While t-SNE is a popular method for this purpose in Euclidean spaces, recent advancements suggest that hyperbolic spaces can better represent hierarchical structures. However, there is a notable lack of data structures and algorithms tailored for hyperbolic spaces. This research addresses this gap by implementing a hyperbolic quadtree structure in the upper half-plane model [7] and integrating it into the hyperbolic t-SNE framework. Our goal is to accelerate the optimization of the hyperbolic t-SNE while maintaining reasonable precision and recall. We conduct rigorous benchmarking experiments to evaluate the performance of this approach, comparing it to existing methods. The findings provide insights into the practical utility of using the hyperbolic quadtree structure in the upper half-plane model in hyperbolic t-SNE embeddings.

1 Introduction

In the dynamic world of data analysis, the exploration and interpretation of high-dimensional datasets have become essential across diverse domains, ranging from sports analytics [20] to machine learning [17] applications. Central to this endeavor is the technique of dimensionality reduction, which facilitates the visualization and comprehension of intricate data. A commonly used method for dimensionality reduction is the t-distributed stochastic neighbor embedding (t-SNE) [11]. This method is popular because it preserves local neighborhoods when embedding the data, as can be seen from the empirical study in Xia et al. [21]. While traditional methods embed data into Euclidean spaces, recent advancements [8], [5] have unveiled the potential of exploring alternative embedding spaces, particularly hyperbolic spaces.

Hyperbolic spaces offer distinct properties that render them ideal for representing hierarchical structures, providing a novel perspective for data analysis compared to the conventional Euclidean counterparts. For instance, it has been shown that it is possible to embed trees into hyperbolic space with arbitrarily low distortion [13], given this property, previous works have shown embeddings of social networks [16] and the Internet [2] in hyperbolic space. However, despite the interest in leveraging hyperbolic spaces for data analysis, there exists a notable void in the development of tailored data structures and algorithms specifically designed for this geometric setting.

Addressing this gap, recent research has proposed a novel quadtree structure tailored for the upper half-plane model [7], offering a promising avenue for efficient spatial partitioning and nearest neighbor search in hyperbolic space. Building upon this foundation, this research aims to implement and utilize this hyperbolic quadtree structure within the context of hyperbolic t-SNE. To motivate the need for accelerated data structures for t-SNE, consider the computational complexity for the t-SNE algorithm. The primary computational bottleneck in t-SNE lies in the calculation of pairwise similarities between data points, which has a time complexity of $O(n^2)$. This quadratic complexity quickly becomes impractical as the number of data points n grows, rendering t-SNE unsuitable for large datasets. The quadtree structure suggested in Kisfaludi-Bak and Wordragen [7] offers building time of $O(n \log(n))$ and reduces the number of computations for similarities by approximating distances efficiently.

The primary objective of this research is twofold: first, to implement the hyperbolic quadtree structure proposed in the aforementioned work [7] and second, to employ it in the framework of hyperbolic t-SNE. By integrating the quadtree structure into the t-SNE framework, we aim to accelerate the optimization of the hyperbolic t-SNE while preserving reasonable precision and recall.

In addition to implementation, this study aims to conduct rigorous benchmarking experiments to evaluate the performance of the hyperbolic quadtree-based approach compared to existing methods, mainly the acceleration suggested in previous work [15].

Contributions

- Implemented a novel quadtree structure tailored for the upper half-plane model in hyperbolic space.
- Integrated this quadtree structure into the hyperbolic t-SNE framework.
- Conducted comprehensive benchmarking experiments to evaluate the performance of the proposed approach.

The remainder of this paper is organized as follows: Section 2 provides essential background information, including an introduction to t-SNE, its Barnes-Hut acceleration structure, key concepts of hyperbolic spaces, the upper half-plane model, and the construction of the hyperbolic quadtree structure. Section 3 reviews related work in the field, highlighting previous research and existing solutions. Section 4 presents the methodology of our proposed solution, detailing the integration of the hyperbolic quadtree structure into the hyperbolic t-SNE framework. Section 5 describes the experimental setup and results, offering insights into the performance evaluation and comparisons with existing methods. Section 6 reflects on the ethical aspects of our research. Finally, Section 7 summarizes our findings, discusses open issues, and suggests directions for future work.

2 Background Information

In this section, we present the techniques and concepts that our method builds upon. We begin with an overview of t-SNE, a popular technique for dimensionality reduction, and its Barnes-Hut acceleration structure for efficient computation in Euclidean spaces. Following this, we explore the essential concepts of hyperbolic spaces, focusing on the upper half-plane model to highlight its unique properties. Finally, we introduce a hyperbolic data structure, the quadtree in the upper half-plane model, which will serve as the foundation for our acceleration of the hyperbolic t-SNE.

2.1 t-distributed Stochastic Neighbor Embedding

The t-SNE is a popular technique for dimensionality reduction, particularly for visualizing high-dimensional data. It works by converting high-dimensional Euclidean distances into conditional probabilities that represent similarities and then mapping these to a lower-dimensional space.

High-dimensional similarities

The similarity between two points x_i and x_j in the high-dimensional space is given by:

$$p_{j|i} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|x_i - x_k\|^2 / 2\sigma_i^2\right)}$$

with the symmetrized joint probabilities:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

here $p_{i|i} = 0$ and σ_i is the variance of the Gaussian centered at point x_i.

Low-dimensional similarities

The similarity in the low-dimensional space is represented using a Student's t-distribution:

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}}$$

where y_i is a point in the low-dimensional embedding corresponding to x_i.

Cost function

t-SNE uses gradient descent to minimize the Kullback-Leibler divergence between the high-dimensional distribution P and the low-dimensional distribution Q:

$$C = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Gradient

The gradient of the Kullback-Leibler divergence with respect to the low-dimensional points y_i is used to minimize the cost function. The gradient is given by:

$$\frac{\delta C}{\delta y_i} = 4 \sum_{j \neq i} (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

This gradient is used in a gradient descent optimization process to update the positions of the low-dimensional points y_i. The naive implementation takes $O(n^2)$ time. This can be seen when rewriting the equation in the following way:

$$\frac{\delta C}{\delta y_i} = 4 \left(\sum_{j \neq i} p_{ij} q_{ij} Z (y_i - y_j) + \sum_{j \neq i} q_{ij}^2 Z (y_i - y_j) \right)$$

Where $Z = \sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}$. We will refer to the first sum as positive forces and the second sum as negative forces. The positive forces can be computed fast if the probability distribution is sparse however the negative forces require $O(n^2)$ time.

t-SNE is effective in preserving local structures [21], making it valuable for visualizing clusters and patterns in high-dimensional data. The method's computational complexity is $O(n^2)$, which is quite slow for large datasets. This prompts the development of acceleration techniques like the Barnes-Hut approximation, which will be discussed next.

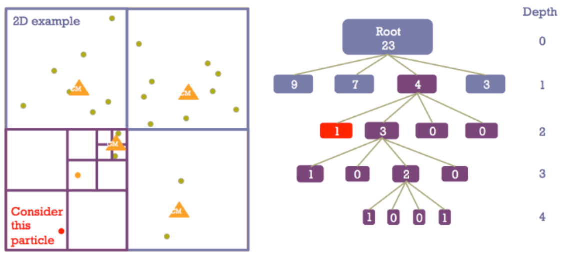

Figure 1: Demonstration of Barnes-Hut approximating with regards to the red point. We can see how the green points are approximated with the middle points (the orange triangles) of the cells. The image was taken from Dierickx and Portillo [3].

2.2 Barnes-Hut Acceleration Structure for t-SNE

The Barnes-Hut algorithm is a hierarchical method that approximates the interaction between distant points to reduce computational complexity. It works by recursively dividing the data space into smaller regions using a quadtree (in two dimensions) or an octree (in three dimensions). Each node in the tree represents a region of space and contains a summary of the points within that region. When evaluating the gradient for a point y_i, we traverse the tree, and at every cell we evaluate if the condition

$$\frac{r_{cell}}{\|y_i - y_{cell}\|} < \theta \qquad (1)$$

holds, where r_{cell} is the diagonal of the cell, y_{cell} is the midpoint of the points in the cell, and θ is a value we chose for the approximation. If the equation holds we do not traverse further but rather approximate all the points in the cell with y_{cell}. This approximation reduces the time complexity from $O(n^2)$ to $O(n \log(n))$ with not much error, given a suitable θ. Normally, θ is set to be between 0.2 and 0.8 [10]. A visualization of the method can be seen in Fig. 1.

2.3 Hyperbolic Space and the upper half-plane model

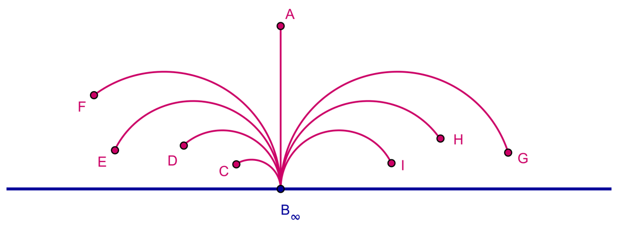

Figure 2: Parallel rays in the half-plane model of hyperbolic geometry [19].

Hyperbolic space is a non-Euclidean geometric space with constant negative curvature, providing a natural framework for representing hierarchical structures and complex networks [2][16]. Unlike Euclidean space, hyperbolic space allows for

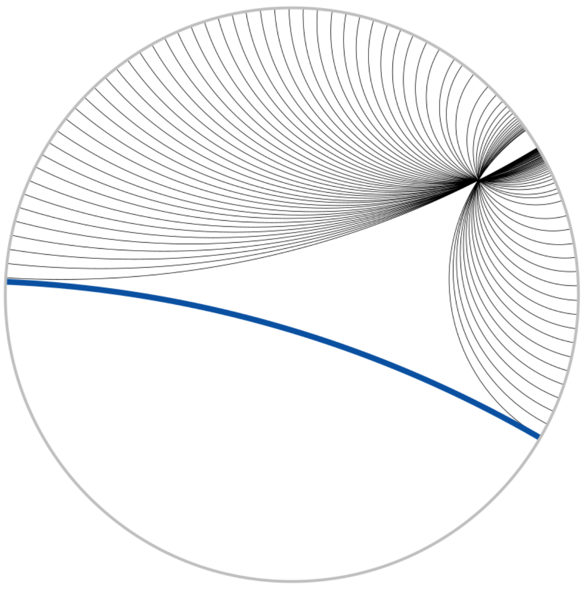

Figure 3: The Poincaré disk model of hyperbolic space with the black lines parallel to the blue line [18].

exponential growth and better captures the relationships in data with intrinsic hierarchical or tree-like properties.

Our data structure is built in the upper half-plane model, see Fig.2. The upper half-plane model similar to the Poincare disk model, see Fig.3, has the conformal property, which helps in splitting the space into hierarchy. The upper half-plane model maps the 2-dimensional hyperbolic space into only the upper half plane (points (x, y) with $x \in \mathbb{R}$ and $y \in \mathbb{R}^+$).

Formally, the upper half-plane model is the space $\mathbb{H} = \{\langle x, y\rangle | y > 0, x, y \in \mathbb{R}\}$ with metric:

$$(ds)^2 = \frac{(dx)^2 + (dy)^2}{y^2}$$

Where s measures the length along a possibly curved line. The hyperbolic distance between two points (x_1, y_1) and (x_2, y_2) in the upper half-plane model is then given by:

$$d^{\mathbb{H}}((x_1, y_1), (x_2, y_2)) = 2arsinh\left(\frac{1}{2}\sqrt{\frac{(x_1-x_2)^2+(z_1-z_2)^2}{z_1 z_2}}\right)$$

2.4 Quadtree in the upper half-plane model

Here we will present the main structure that we will use for accelerating our computations.

Horobox

In Kisfaludi-Bak and Wordragen [7], horoboxes are introduced as the hyperbolic counterpart to Euclidean axis-parallel boxes in a fixed half-space model. These horoboxes are defined by their corner points, $(x_{\min}(B), z_{\downarrow}(B))$ and $(x_{\max}(B), z_{\uparrow}(B))$, which are minimal and maximal in all coordinates, respectively.

Definition. In a fixed half-space model, a cube-based horobox C is an axis-parallel horobox with $\frac{z_{\uparrow}(C)}{z_{\downarrow}(C)} = 2^h$ and $\frac{x_{max}(C)-x_{min}(C)}{z_{\downarrow}(C)} = w$, where $w = w(C)$ is called the width and $h = h(C)$ is called the height of the horobox.

Horoboxes will be the cells of our tree. For a horobox C, point (p_x, p_z) will be inside C if $x_{min}(C) \leq p_x < x_{max}(C)$ and $z_{\downarrow}(C) \leq p_z < z_{\uparrow}(C)$.

Initializing the root cell

For a given set of points $P = \{(p_{x_1}, p_{z_1}), (p_{x_2}, p_{z_2}), ..., (p_{x_n}, p_{z_n})\}$. First, we find the bounding horobox C' of P. The bounding horobox is defined by the minimal and maximal coordinates of all the given points in the x and z directions. Formally, $x_{min}(C') = \min_{1\leq i\leq n} p_{x_i}$, $x_{max}(C') = \max_{1\leq i\leq n} p_{x_i}$, and anlogically $z_{\downarrow}(C') = \min_{1\leq i\leq n} p_{z_i}$ and $z_{\uparrow}(C') = \max_{1\leq i\leq n} p_{z_i}$. Then if $w(C') \leq 1$ and $h(C') \leq 1$, we find the smallest $\ell \in \mathbb{Z}$ such that $w(C') \leq 2^{\ell}$ and $h(C') \leq 2^{\ell}$ then we let $w(C) = 2^{\ell}$ and $h(C) = 2^{\ell}$. Otherwise, we find the smallest $\ell \in \mathbb{Z}$ such that $w(C') \leq 2^{2^{\ell}-1}$ and $h(C') \leq 2^{\ell}$, then we let $w(C) = 2^{2^{\ell}-1}$ and $h(C) = 2^{\ell}$.

After that we let $z_{\downarrow}(C) = z_{\downarrow}(C')$ and $x_{\min}(C) = x_{\min}(C')$. Following the definition of a horobox after obtaining $h(C)$ and $w(C)$, we set $z_{\uparrow}(C) = z_{\downarrow}(C) \times 2^{h(C)}$ and $x_{max}(C) = w(C) \times z_{\downarrow}(C) + x_{min}(C)$. C will be the root cell of our tree and ℓ indicates the level of the cell. Given the way we initialized C, it can be easily seen that $z_{\uparrow}(C') \leq z_{\uparrow}(C)$ and $x_{max}(C') \leq x_{max}(C)$ and since C' contained all the points in P, therefore, C will also contain P.

Spliting Criteria

For a cell C^* if $h(C^*) \leq 1$ we split it into 4 cells along the axis-parallel lines $(\frac{x_{min}+x_{max}}{2}, \sqrt{z^{\downarrow}(C^*)z^{\uparrow}(C^*)})$, if $h(C^*) > 1$ first we split it along the $z = \sqrt{z^{\downarrow}(C^*)z^{\uparrow}(C^*)}$. This gives us two horoboxes with height $\frac{h(C^*)}{2}$, where the top one has width $w(C^*)/2^{\frac{h(C^*)}{2}}$ but the bottom one still $w(C^*)$. Thus, we split the bottom cell into $2^{\frac{h(C^*)}{2}}$ cells each having the same width $w(C^*)/2^{\frac{h(C^*)}{2}}$. We continue splitting the cells while there are still points in them. A child node will only contain the points from its parent that are within the bounds of its box. In Fig. 4 we can see a visualization of the tree.

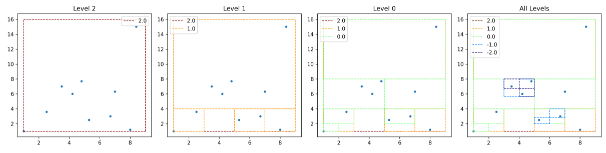

Figure 4: An example of building the tree, the first plot shows the root cell, the second and third show the next two levels, and the last plot shows the whole tree.

Properties of the quadtree

Here we will present some properties of the quadtree that are proven in Kisfaludi-Bak and Wordragen [7].

- At any level ℓ, cells are cube-based horoboxes with height 2^{ℓ}.
- For $\ell \geq 0$, the width is $2^{2^{\ell}-1}$ and the diameter is $2arsinh(2^{2^{\ell}-2})$.
- For $\ell < 0$ the width is $\alpha 2^{\ell}$ and the diameter is $2arsinh(\frac{1}{2}\sqrt{\frac{\alpha^2 4^{\ell}+(2^{2^{\ell}}-1)^2}{2^{2^{\ell}}}})$, where α is a cell-specific value $\in (1/2, 1)$
- Cells of the same level $\ell \geq 0$ are isometric, and cells of level $\ell < 0$ are cube-based horoboxes with the same height whose width differs by less than a factor two.

These properties offer us a promising structure to replace the Barnes-Hut in the hyperbolic space. The bulding of this structure given n points can be done in $O(n\log(n))$. Approximating the negative forces in the gradient can also be done in $O(n\log(n))$ time, given an appropriate θ. The results of the benchmarking of the building and the calculation of the negative forces can be seen in Section 5.2 and Section 5.2.

2.5 Einstein Midpoint

In the Barnes-Hut method, each cell has a center, which is taken to be the arithmetic mean of all the points in the cell, this mean is not available in hyperbolic space. In hyperbolic space, we can instead take the Einstein midpoint of the points in the cell. The Einstein midpoint is in the Klein model and is calculated as follows:

$$m(\{v_j\}) = \sum_j \left(\frac{\gamma(v_j)}{\sum_l \gamma(v_l)}\right) v_j$$

where $\gamma(v_j) = \frac{1}{\sqrt{1-\|v_j\|^2}}$ and v_j are the coordinates of y_j in the Klein model. This point is suitable because it can be computed in $O(n)$ time for n points. This means that if our tree takes n points at each level this will add $O(n)$ time complexity. Since the number of levels of our tree is approximated to be $O(\log(n))$, therefore for the whole construction of the tree this will add $O(n\log(n))$ time which will not affect the overall complexity of building the tree $O(n\log(n))$.

3 Related work

3.1 Dimensionality Reduction Methods

Dimensionality reduction techniques can be categorized based on whether they use linear or non-linear embeddings and whether they aim to preserve local or global distances. In this work, we focus on t-SNE, a non-linear, locally preserving method. Other methods in this category include Locally Linear Embedding (LLE) [14], Laplacian Eigenmaps (LE) [1], Local Affine Multidimensional Projection (LAMP) [6], and Uniform Manifold Approximation and Projection (UMAP) [12]. Comprehensive surveys detail the advantages and drawbacks of these techniques, often highlighting t-SNE's superior performance in clustering tasks [21].

3.2 Hyperbolic Embeddings

Embedding data into hyperbolic space has shown significant potential, particularly for representing hierarchical structures inherent in many real-world graphs and networks. Studies demonstrate that trees can be embedded into two-dimensional hyperbolic space with low distortion [13]. This property extends to various real-world networks, such as embedding the Internet [2] and social networks [16] in hyperbolic space.

3.3 Hyperbolic t-SNE Extensions

Several extensions of t-SNE to hyperbolic spaces have been introduced to better capture the hierarchical nature of high-dimensional data. Notable among these are Cauchy Origin-SNE (CO-SNE) [4] and Poincarè maps [8]. CO-SNE employs a Riemannian normal distribution for high-dimensional data and a Cauchy distribution for low-dimensional probabilities, adding terms to the cost function to maintain hierarchical structures. Poincarè maps create a nearest-neighbor graph and utilize Gaussian kernels for probability distributions while employing a symmetric Kullback-Leibler divergence as the cost function.

3.4 Accelerating t-SNE

Acceleration methods for t-SNE, such as the Barnes-Hut approximation, see Section 2.2, and Fourier transform-based approach [9], significantly reduce the computational complexity in Euclidean spaces. However, these methods face challenges when adapted to hyperbolic spaces due to the non-linear nature and unique geometric properties of hyperbolic space. Previous work [15] has shown a quadtree in the Poincarè disk model that significantly accelerates the hyperbolic t-SNE. Our research will build on the implementation of Skrodzki et al. [15] by implementing a hyperbolic quadtree structure in the upper half-plane model [7].

4 Methodology

This section provides a detailed explanation of the methodology used in our research. We implemented the suggested quadtree in the upper half-plane model (see Section 2.4) and integrated it into the existing implementation of hyperbolic t-SNE, which originally works for the Poincarè disk model [15]. The following steps outline our approach to modifying the hyperbolic t-SNE algorithm and incorporating our hyperbolic quadtree in the upper half-plane model for efficient computation. The diagram in Fig. 5 shows the steps we make at each step of the gradient descent.

4.1 Transformation Between Models

At each step of the gradient descent, we transformed the points from the Poincarè disk model to the upper half-plane model. This transformation is essential for utilizing the quadtree in the upper half-plane. A point (x, y) in the Poincarè disk model maps to $\left(\frac{2x}{x^2+(1-y)^2}, \frac{1-x^2-y^2}{x^2+(1-y)^2}\right)$ in the upper half-plane model. Conversely, a point (x, y) in the upper half-plane model maps to $\left(\frac{2x}{x^2+(1+y)^2}, \frac{x^2+y^2-1}{x^2+(1+y)^2}\right)$ in the Poincarè disk model. The formulas are taken from Wikipedia contributors [18].

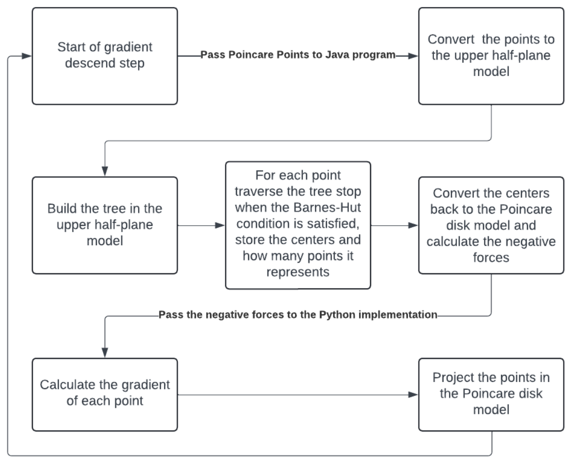
Start of gradient descend step
Pass Poincare Points to Java program
Convert the points to the upper half-plane model
Build the tree in the upper half-plane model
For each point traverse the tree stop when the Barnes-Hut condition is satisfied, store the centers and how many points it represents
Convert the centers back to the Poincare disk model and calculate the negative forces
Pass the negative forces to the Python implementation
Calculate the gradient of each point
Project the points in the Poincare disk model

Figure 5: Diagram of how the gradient descend step works for our method.

4.2 Quadtree Construction

After transforming the points to the upper half-plane model we construct our quadtree as discussed in Section 2.4. Each cell's center was computed using the Einstein midpoint of the points in it to ensure a reasonable approximation for the points in the cell, see Section 2.5. A visualization of the tree can be seen in Fig. 4.

4.3 Negative forces calculation and Integration

For each point, we traverse our tree in a similar manner as Barnes-Hut (see Section 2.2), we stop when the condition in (1) is satisfied and approximate each point in the cell with the cell's center (the Einstein midpoint of the points in the cell). However, note that the Euclidian distance between the point and the cell's Einstein midpoint should be converted to distance in the upper half-plane model. With these approximations, we then calculate the negative forces for each point.

4.4 Projection Back to the Poincarè Disk Model

After the gradient was calculated, we projected the points back to the Poincarè disk model. This ensured compatibility with the original t-SNE implementation and allowed us to leverage existing visualization tools.

4.5 Implementation Details

Our implementation of the quadtree is written in JAVA for our convenience. The integration with the existing Python and Cython implementation of hyperbolic t-SNE is achieved through file-based data exchange. Specifically, the Python code writes the Poincarè coordinates of the points and the theta value to a file at each step of the gradient descent, then the Python code runs our JAVA implementation. The JAVA program then reads these files, converts the points to the upper half-plane model, builds the quadtree, calculates the negative forces, and writes the results to a file. Finally, the Python program reads the negative forces from the file and uses them to calculate the gradients of the points. We chose to use file-based data exchange for ease of implementation, though we acknowledge that this method is slower due to the overhead of file I/O operations.

5 Experimental Setup and Results

In this section, we describe the experimental setup used to evaluate the performance of our hyperbolic quadtree-based t-SNE approach. We detail the datasets, simulation environment, and metrics used. Subsequently, we present the results, including precision, recall, the effect of the parameter θ, and the time required to build the quadtree.

5.1 Experimental Setup

Datasets

We used publicly available datasets for our experiments, ensuring a wide range of data characteristics and complexities, see Table.1.

Name	Data Type	# Points	# Dim.	# Cl.
PLANARIA	single-cell	21,612	50	51
MNIST	images	70,000	784	10
C_ELEGANS	single cell	89,701	20,222	37

Table 1: Data sets used in the experiments with the number of points, the dimension, and the number of labeled classes.

Simulation Environment

The experiments were conducted on a HP ZBook Power G7:

- **Processor**: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz
- **Memory**: 16,0 GB

Evaluation Metrics

To evaluate the performance of our approach, we used the following metrics:

- **Precision and Recall**: To measure the accuracy of neighborhood preservation. Refer to the evaluation in Skrodzki et al. [15].
- **Effect of θ**: To analyze how different values of the parameter θ impact the performance.
- **Tree Construction Time**: To assess the efficiency of building the quadtree in the upper half-plane model.
- **Negative forces calculation time**: To assess the efficiency of the summarizing our tree does.

5.2 Results

Precision and Recall

We calculated precision and recall to evaluate how well the local neighborhood structure is preserved by our method compared to the traditional quadtree in the Poincarè disk model, and the exact method. Following Skrodzki et al. [15], to generate the precision and recall curve we did the following:

We fixed a maximum neighborhood size $k_{max} = 30$. Then, for each $k \in \{1, 2, ..., k_{max}\}$, we computed the number of true positives as $TP_k = N_{k_{max}}(X) \cap N_k(Y)$, that

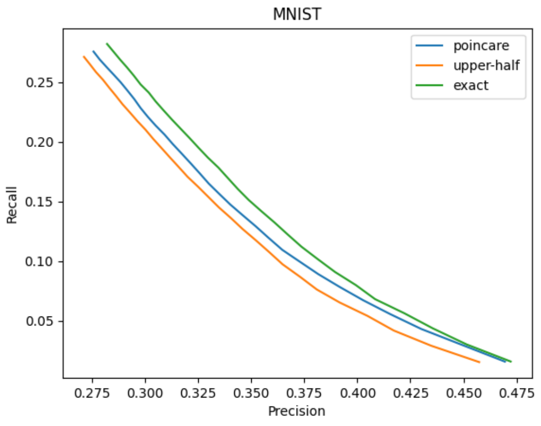

Figure 6: A plot comparing the precision-recall of our quadtree in the upper half-plane model, the quadtree in the Poincarè disk model, and the exact method with 10000 samples from the MNIST dataset

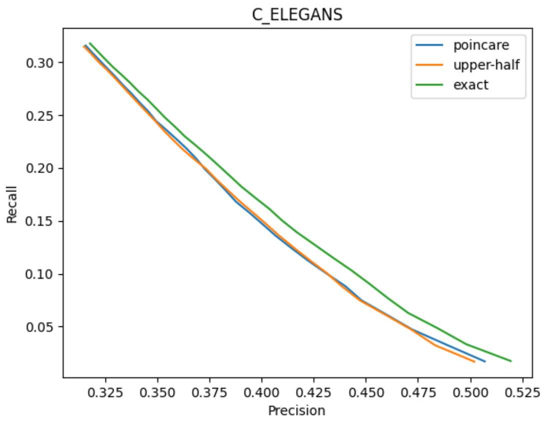

Figure 7: A plot comparing the precision-recall of our quadtree in the upper half-plane model, the quadtree in the Poincarè disk model, and the exact method with 10000 samples from the C-ELEGANS dataset

is the points that are in the high-dimensional neighborhood as well as in the low-dimensional neighborhood, given the respective metric. From here, we obtain the precision as $PR_k = \frac{|TP_k|}{k}$, and the recall as $RC_k = \frac{TP_k}{k_{max}}$.

The results, see Figs.6, 7, and 8 demonstrated that our approach maintains high precision and recall values, however, it does not outperform the existing solution based on a quadtree in the Poincarè disk model. Furthermore, the embeddings obtained from both methods, see Fig.9, are quite similar, which serves as a proof of concept.

Effect of θ

The parameter θ controls the trade-off between accuracy and computational efficiency in the Barnes-Hut approximation. We varied θ from $(0.0, 0.1, 0.2..., 1.0)$ and observed its effect on precision, recall, and computation time. The results, illustrated in Figure 10, show that generally smaller θs produce better precision-recall curves, however, note that $\theta = 0.8$ and $= 0.7$ produce the best precision-recall curves, which means that generally the best θ depends on the sample. The computational time for each step of the gradient descent can be seen in Fig. 11. This shows that higher θs approximate more and run faster. Furthermore, when $\theta = 0$ the method doesn't do any approximations and arguably should be like the exact solution with $O(n^2)$ time complexity, when we increase the θ we can see how the runtime decreases drastically, which shows the efficiency of our method.

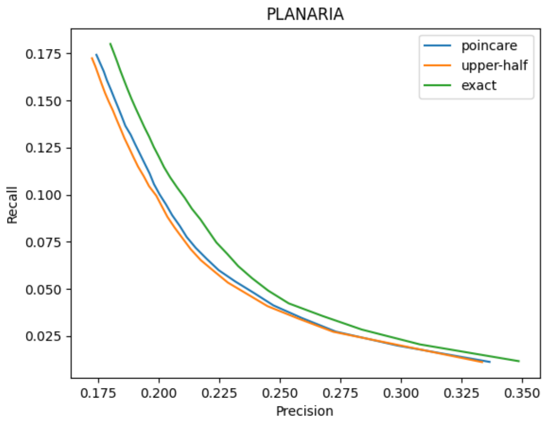

Figure 8: A plot comparing the precision-recall of our quadtree in the upper half-plane model, the quadtree in the Poincarè disk model, and the exact method with 10000 samples from the PLANARIA dataset

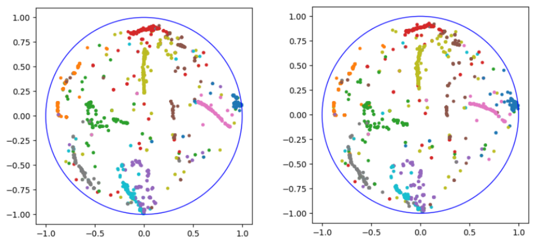

Figure 9: On the left we can see the final embedding obtained when using our proposed method, and on the right is the embedding after using the quadtree in the Poincarè disk model. This is the MNIST dataset with 10000 samples.

Tree Construction Time

We measured the time required to build the quadtree in the upper half-plane model for different sample sizes. The results, presented in Table 2, indicate that our approach scales efficiently with the size of the sample.

Negative forces calculation time

We measured the time our method takes to calculate the negative forces used in the derivative for all the points. The results can be seen in Table 3. This shows that our method scales well with the size of the sample.

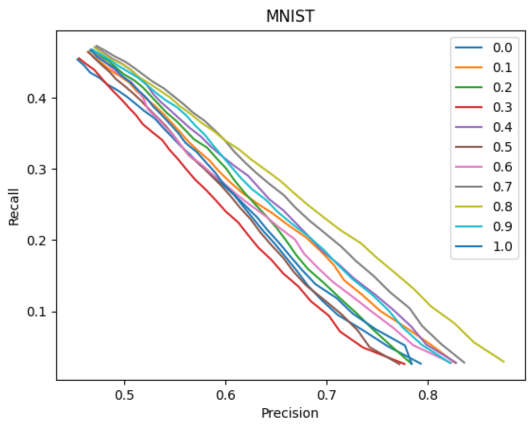

Figure 10: A plot comparing the precision-recall of our method given different θs from $(0, 0.1, 0.2, ..., 1)$. This is the MNIST dataset with 10000 samples.

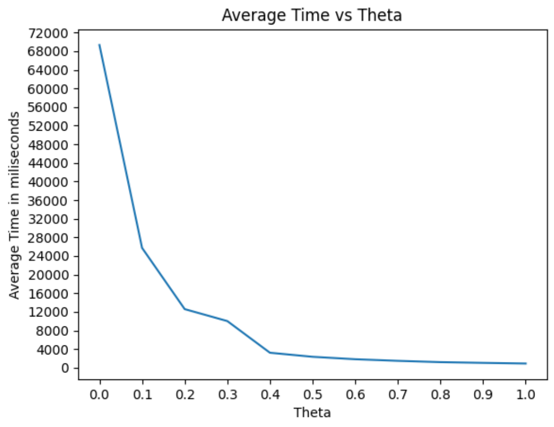

Figure 11: A plot comparing the average time to calculate the negative forces of our method given different θs from $(0, 0.1, 0.2, ..., 1)$. This is the MNIST dataset with 10000 samples.

5.3 Discussion

The results demonstrate that our hyperbolic quadtree-based t-SNE approach effectively preserves local neighborhood structures with high precision and recall, while also being computationally efficient. The parameter θ allows for a flexible trade-off between accuracy and performance, and our method scales well with increasing dataset sizes. These findings suggest that the proposed approach is a viable and effective method for dimensionality reduction in hyperbolic spaces.

6 Responsible Research

In conducting this research, several ethical considerations were taken into account to ensure the integrity and reliability of our findings. This section reflects on these aspects, providing a comprehensive overview of our commitment to responsible research practices.

Sample Size	Construction Time
1,000	29.95 ms
5,000	83.55 ms
10,000	120 ms
20,000	187.33 ms
40,000	308.32 ms

Table 2: Quadtree construction time for different sample sizes of the MNIST dataset. We ran the gradient descent for each sample size for around 150 steps and took the average time.

Sample Size	Negative forces time calculation
1,000	254.75 ms
5,000	1043.28 ms
10,000	2275.40 ms
20,000	5031.23 ms
40,000	12534.54 ms

Table 3: Time for calculation of the negative forces for different sample sizes of the MNIST dataset. We ran the gradient descent for each sample size for around 150 steps and took the average time.

6.1 Data Privacy and Confidentiality

One of the primary ethical concerns in data analysis is the handling of data privacy and confidentiality. Our research involves the use of publicly available high-dimensional datasets, which are already anonymized and devoid of any personally identifiable information (PII). Using these publicly available datasets ensures that our study adheres to ethical standards without compromising data privacy.

6.2 Transparency and Openness

Transparency is crucial in scientific research to build trust and facilitate verification of results. In line with this principle, we will make our code publicly available by adding it to a TU Delft repository. This allows other researchers to replicate our experiments, verify our findings, and build upon our work.

6.3 Use of AI Tools

To enhance the clarity and formality of this report, we utilized ChatGPT, an AI-based language model, primarily for paraphrasing and refining our written content. This tool helped us ensure that the document is well-articulated and accessible to a broad audience. However, all the technical content, data analysis, and conclusions presented in this report are the result of our original research efforts.

7 Conclusions and Future Work

In this study, we tackled the challenge of dimensionality reduction in hyperbolic spaces by implementing a novel hyperbolic quadtree structure within the upper half-plane model and integrating it into the hyperbolic t-SNE framework. Our primary objectives were to enhance the preservation of local neighborhood structures in high-dimensional data and to evaluate the performance of this approach through rigorous benchmarking experiments.

Our findings indicate that the proposed hyperbolic quadtree structure is computationally efficient, maintaining high precision and recall values. However, it does not significantly outperform the existing solution based on the quadtree in the Poincarè disk model. The embeddings obtained from both methods are quite similar, which serves as a proof of concept that our method is viable but not necessarily superior in terms of precision and recall.

Future Work

To further enhance the effectiveness and efficiency of hyperbolic t-SNE embeddings, we recommend the following directions for future research:

- **Implementation in C++**: To reduce the overhead associated with writing the points and the negative forces to files and reading back, the entire JAVA implementation can be developed in C++. This would potentially streamline the process and enhance performance.
- **Broader Applications**: Applying our method to a wider range of datasets could provide deeper insights into its generalizability and robustness.
- **Comparative Analysis**: Conducting more extensive comparative analyses with other state-of-the-art methods for hyperbolic embeddings could further demonstrate the strengths and limitations of our approach.
- **Exploring Other Models**: Investigating other models of hyperbolic space, beyond the upper half-plane and Poincarè disk, might reveal alternative approaches that offer better performance or new insights into hyperbolic embeddings.

In conclusion, this research provides a foundational step towards efficient dimensionality reduction in hyperbolic spaces, highlighting the potential and limitations of using a quadtree structure in the upper half-plane model. While our approach is effective, further refinements and optimizations are necessary to fully capitalize on the benefits of hyperbolic space for high-dimensional data analysis.

References

[1] Mikhail Belkin and Partha Niyogi. "Laplacian eigenmaps for dimensionality reduction and data representation". In: *Neural computation* 15.6 (2003), pp. 1373–1396.

[2] Marian Boguna, Fragkiskos Papadopoulos, and Dmitri Krioukov. "Sustaining the Internet with hyperbolic mapping". In: *Nature Communications* 1.1 (Sept. 2010). ISSN: 2041-1723. DOI: 10.1038/ncomms1063. URL: http://dx.doi.org/10.1038/ncomms1063.

[3] Marion Dierickx and Stephen Portillo. *N-Body Building*. http://portillo.ca/nbody/barnes-hut/. Accessed: 11-June-2024.

[4] Yunhui Guo, Haoran Guo, and Stella X Yu. "Co-sne: Dimensionality reduction and visualization for hyperbolic data". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 21–30.

[5] Yunhui Guo, Haoran Guo, and Stella X. Yu. "CO-SNE: Dimensionality Reduction and Visualization for Hyperbolic Data". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 11–20. DOI: 10.1109/CVPR52688.2022.00011.

[6] Paulo Joia et al. "Local Affine Multidimensional Projection". In: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (2011), pp. 2563–2571. DOI: 10.1109/TVCG.2011.220.

[7] Sándor Kisfaludi-Bak and Geert van Wordragen. *A Quadtree, a Steiner Spanner, and Approximate Nearest Neighbours in Hyperbolic Space*. 2023. arXiv: 2305.01356 [cs.CG].

[8] A. Klimovskaia et al. "Poincaré maps for analyzing complex hierarchies in single-cell data". In: *Nature Communications* 11.1 (2020), pp. 1–9. DOI: 10.1038/s41467-020-16822-4.

[9] George C Linderman et al. "Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data". In: *Nature methods* 16.3 (2019), pp. 243–245.

[10] Laurens van der Maaten. "Accelerating t-SNE using Tree-Based Algorithms". In: *Journal of Machine Learning Research* 15.93 (2014), pp. 3221–3245. URL: http://jmlr.org/papers/v15/vandermaaten14a.html.

[11] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

[12] Leland McInnes, John Healy, and James Melville. "Umap: Uniform manifold approximation and projection for dimension reduction". In: *arXiv preprint arXiv:1802.03426* (2018).

[13] Rik Sarkar. "Low Distortion Delaunay Embedding of Trees in Hyperbolic Plane". In: Sept. 2011, pp. 355–366. ISBN: 978-3-642-25877-0. DOI: 10.1007/978-3-642-25878-7_34.

[14] Lawrence K Saul and Sam T Roweis. "An introduction to locally linear embedding". In: *unpublished. Available at: http://www. cs. toronto. edu/~roweis/lle/publications. html* (2000).

[15] Martin Skrodzki et al. *Accelerating hyperbolic t-SNE*. 2024. arXiv: 2401.13708 [cs.HC].

[16] Kevin Verbeek and Subhash Suri. "Metric Embedding, Hyperbolic Space, and Social Networks". In: *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*. SOCG'14. Kyoto, Japan: Association for Computing Machinery, 2014, pp. 501–510. ISBN: 9781450325943. DOI: 10.1145/2582112.2582139. URL: https://doi.org/10.1145/2582112.2582139.

[17] Junpeng Wang et al. "Visual Analytics for RNN-Based Deep Reinforcement Learning". In: *IEEE Transactions on Visualization and Computer Graphics* 28.12 (2022), pp. 4141–4155. DOI: 10.1109/TVCG.2021.3076749.

[18] Wikipedia contributors. *Poincaré disk model — Wikipedia, The Free Encyclopedia*. [Online; accessed 11-June-2024]. 2024. URL: https://en.wikipedia.org/w/index.php?title=Poincar%C3%A9_disk_model&oldid=1211324702.

[19] Wikipedia contributors. *Poincaré half-plane model — Wikipedia, The Free Encyclopedia*. [Online; accessed 11-June-2024]. 2023. URL: https://en.wikipedia.org/w/index.php?title=Poincar%C3%A9_half-plane_model&oldid=1155587333.

[20] Jiang Wu et al. "TacticFlow: Visual Analytics of Ever-Changing Tactics in Racket Sports". In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (2022), pp. 835–845. DOI: 10.1109/TVCG.2021.3114832.

[21] Jiazhi Xia et al. "Revisiting Dimensionality Reduction Techniques for Visual Cluster Analysis: An Empirical Study". In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (Jan. 2022), pp. 529–539. ISSN: 2160-9306. DOI: 10.1109/tvcg.2021.3114694. URL: http://dx.doi.org/10.1109/TVCG.2021.3114694.