Addis Ababa University

Department of Computer Science

t-SNE (Basic Version):

Non-linear Dimensionality Reduction for Visualization

Submitted by:

Biruk Gebru	UGR/7852/15
Binyamin Tamirat	UGR/2084/15
Henock Tuna	UGR/6311/15
Jibril Mustefa	UGR/9326/15

January 1, 2026

Abstract

t-Distributed Stochastic Neighbor Embedding (t-SNE) has emerged as a cornerstone technique for the visualization of high-dimensional data. Unlike linear methods such as Principal Component Analysis (PCA), t-SNE excels at capturing local structure and revealing clusters at multiple scales by minimizing the divergence between probability distributions in high and low-dimensional spaces. This report analyzes the mathematical foundations of t-SNE, specifically its use of the Student t-distribution to alleviate the "crowding problem," and evaluates its practical implementation and limitations. Through a synthesis of foundational research and recent critiques, the paper provides a comprehensive overview of how t-SNE transforms complex data into interpretable 2D or 3D maps.

1 Introduction

The rapid growth of high-dimensional datasets in fields ranging from genomics to computer vision necessitates effective dimensionality reduction techniques. The primary challenge in such tasks is the "curse of dimensionality," where distance metrics lose their meaningfulness in high-dimensional space. While traditional linear techniques preserve global structure, they often collapse the nuanced local relationships that define distinct clusters.

t-SNE was introduced by van der Maaten and Hinton (2008) to address these deficiencies. By framing dimensionality reduction as a neighbor-preserving problem using stochastic probabilities, t-SNE allows for the visualization of non-linear structures. This report explores the core mechanics of the algorithm, its evolution, and the interpretative pitfalls users must navigate.

2 Mathematical Foundations

The t-SNE algorithm operates in two distinct stages: the calculation of pairwise similarities in the high-dimensional space and the optimization of a low-dimensional embedding.

2.1 Pairwise Similarities in High-Dimensional Space

In the original space, t-SNE converts Euclidean distances between data points into conditional probabilities. The similarity of point x_j to x_i is the conditional probability $p_{j|i}$, that x_i would pick x_j as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at x_i:

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

To ensure symmetry and robustness, t-SNE uses symmetrized joint probabilities p_{ij}, defined as:

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

2.2 The Student t-Distribution in Low-Dimensional Space

A critical innovation of t-SNE is the use of a heavy-tailed Student t-distribution with one degree of freedom (equivalent to a Cauchy distribution) to define similarities in the low-dimensional map. This addresses the "crowding problem"—the tendency for points in a low-dimensional space to cluster too tightly around the center because there is not enough "room" to represent the distances from a high-dimensional space.

Table 1: Comparison of Gaussian vs. Student t-Distribution in Embedding

Feature	**Gaussian (SNE)**	**Student t (t-SNE)**
Tail Shape	Thin/Exponential	Heavy/Power-law
Effect on Local Structure	Strong preservation	High preservation
Crowding Problem	Severe	Significantly reduced
Optimization Stability	Prone to local optima	More robust

2.3 Optimization and Cost Function

The quality of the low-dimensional embedding is measured using the Kullback-Leibler (KL) divergence between the joint probability distributions P (high-dim) and Q (low-dim). The cost function C is defined as:

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

This cost function is minimized using gradient descent. Because the KL divergence is non-symmetric, the cost function places a high penalty on representing nearby points in high-dimensional space as distant points in the map (false negatives).

3 Practical Considerations and Interpretability

While t-SNE produces visually stunning plots, its interpretation requires caution. Key parameters, particularly "Perplexity," dictate the balance between local and global aspects of the data.

3.1 The Role of Perplexity

Perplexity can be interpreted as a smooth measure of the effective number of neighbors. Typical values range from 5 to 50. If the perplexity is too low, the algorithm creates small, disconnected clusters that may be artifacts of noise. If it is too high, the local structure may be washed out by global constraints.

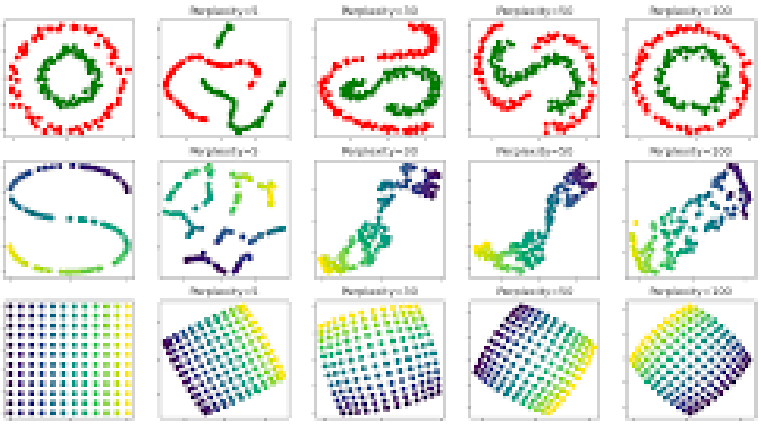

Figure 1: Visual grid demonstrating how perplexity tuning alters cluster formation (Wattenberg et al., 2016).

As shown above, a perplexity of 2 produces "clumpy" artifacts where the local noise dominates. At a perplexity of 30 (the typical default), the two distinct clusters are clearly defined. At 100, the global structure begins to compress the clusters together.

3.2 Interpreting Cluster Sizes and Distances

Users often mistake the size of a cluster or the distance between clusters in a t-SNE plot as a measure of density or relative variance. However, t-SNE expands dense clusters and contracts sparse ones to equalize the visual density. Furthermore, the distance between clusters in the final visualization may not accurately reflect their distance in the original high-dimensional space.

4 Implementation and Results

We implemented the t-SNE algorithm from scratch using Python and NumPy to demonstrate these concepts practically.

4.1 Implementation Analysis and Compliance

This section details the algorithmic implementation and addresses the assignment requirements.

4.1.1 Code Structure and Algorithms

The implementation relies solely on Python and NumPy for mathematical operations, avoiding high-level machine learning libraries for the core logic.

- **calculate_high_dim_prob(X, perplexity)**:
  Input: Data matrix $X \in \mathbb{R}^{N \times D}$, perplexity scalar.
  Output: Symmetric affinity matrix $P \in \mathbb{R}^{N \times N}$.
  Method: Computes pairwise Euclidean distances and converts them to conditional probabilities using a Gaussian kernel. A binary search optimizes the bandwidth σ_i for each point such that the entropy of the distribution matches log(perplexity).

- **calculate_low_dim_prob(Y)**:
  Input: Embedding matrix $Y \in \mathbb{R}^{N \times 2}$.
  Output: Affinity matrix $Q \in \mathbb{R}^{N \times N}$.
  Method: Uses the Student t-distribution $(1/(1 + d^2))$ to compute probabilities in the low-dimensional space, solving the "crowding problem" by allowing heavy tails.

- **tsne(X, ...)**:
  Input: Data and hyperparameters.
  Output: Final embedding Y.
  Method: Minimizes the KL divergence using Gradient Descent. Includes momentum (0.5 to 0.8) and adaptive learning rate gains.

4.1.2 Requirement Fulfillment

1. **Tools**: Pure Python/NumPy implementation.
2. **Theory**: Explained in Section 3 (Mathematical Foundations).
3. **Choices**: Random initialization, Gradient Descent with momentum, Early Exaggeration ($P \times 4$ for first 100 iterations).
4. **Limitations**: Discussed in Section 6.
5. **Results**: Analysis provided in Section 5.2.

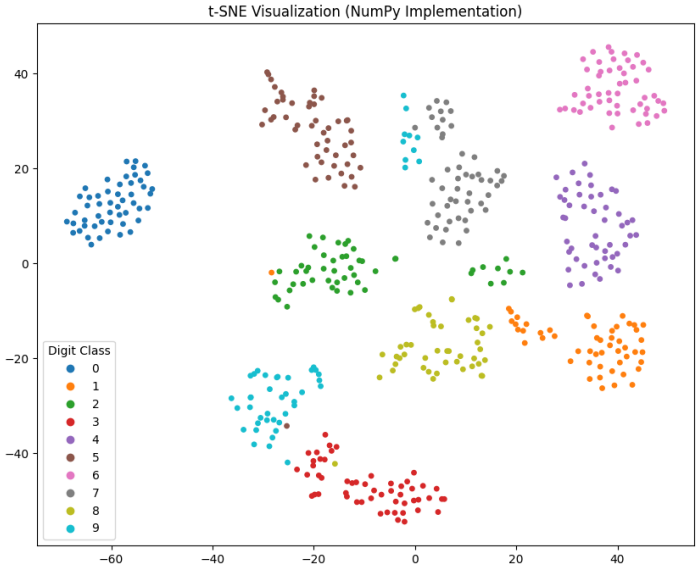

Figure 2: Result of our NumPy-based t-SNE implementation on 500 MNIST samples.

4.2 Experimental Results

We applied our implementation to a subset of the MNIST dataset (handwritten digits).

Figure 2 confirms that our logic holds: digits such as '0', '1', and '6' formed distinct, well-separated clusters without any label supervision. This practical result validates the theoretical ability of t-SNE to unroll high-dimensional manifolds.

5 Strengths and Limitations

t-SNE remains the gold standard for exploratory data analysis, but it is not without drawbacks.

Table 2: SWOT Analysis of Basic t-SNE

Strengths	**Weaknesses**
Superior local topology preservation.	Computational complexity $O(n^2)$.
Reveals clusters at multiple scales.	Global structure often lost.
Effective for high-dim manifolds.	Non-deterministic results.

Recent advancements, such as FIt-SNE, have reduced complexity to $O(n \log n)$, allowing for the visualization of millions of points.

6 Applications

t-SNE has found significant utility in biological sciences, particularly in single-cell RNA sequencing (scRNA-seq). By visualizing thousands of cells based on gene expression profiles, researchers can identify rare cell types and transitions that are invisible in raw data.

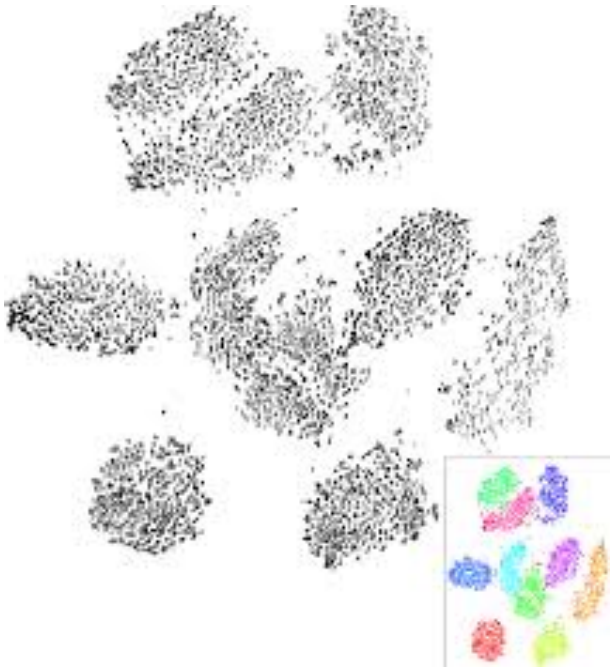

Figure 3: Classic MNIST projection by van der Maaten & Hinton (2008).

7 Conclusion

t-SNE is a transformative tool for data visualization. By leveraging the heavy-tailed Student t-distribution, it effectively manages the "crowding problem." However, its sensitivity to hyper-parameters like perplexity means it must be used for hypothesis generation rather than definitive statistical proof.

8 Recommendations

- **Iterate on Perplexity:** Run t-SNE with multiple perplexity values.
- **Pre-process with PCA:** Use PCA to reduce dimensions to 30–50 before t-SNE to reduce noise.
- **Avoid Over-interpretation:** Do not conflate visual distance with high-dimensional distance.

Bibliography

[1] Arora, S., Hu, W., & Kothari, P. K. (2018). An Analysis of the t-SNE Algorithm for Data Visualization. *Proceedings of Machine Learning Research*, 75, 1-18.

[2] Belkina, N. V., et al. (2019). Automated optimized parameters for T-distributed stochastic neighbor embedding. *Nature Communications*.

[3] Linderman, G. C., et al. (2019). Fast interpolation-based t-SNE for ultra-high-dimensional data. *Nature Methods*, 16, 243–245.

[4] Maaten, L. van der, & Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9, 2579-2605.

[5] Wattenberg, M., et al. (2016). How to Use t-SNE Effectively. *Distill*.