t-SNE Tutorial

September 17, 2025

1 Introduction

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a non-linear dimensionality reduction technique for visualizing high-dimensional data in two or three dimensions. It converts pairwise distances into probabilities and minimizes the Kullback–Leibler (KL) divergence between neighborhoods in the original and embedded spaces, yielding intuitive visual clusters.

2 Theory and Formulas

2.1 Pairwise Similarities

For high-dimensional points $\mathbf{x}_i$, t-SNE defines conditional probabilities that measure neighbor affinity:

$$p_{j|i} = \frac{\exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 / 2\sigma_i^2\right)}{\sum_{k \neq i} \exp\left(-\|\mathbf{x}_i - \mathbf{x}_k\|_2^2 / 2\sigma_i^2\right)}, \tag{1}$$

where σ_i is set so that the perplexity of the distribution equals a user-specified value.

2.2 Low-Dimensional Embedding

In the embedding $\mathbf{y}_i$, t-SNE uses a heavy-tailed Student t-distribution with one degree of freedom:

$$q_{ij} = \frac{\left(1 + \|\mathbf{y}_i - \mathbf{y}_j\|_2^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|\mathbf{y}_k - \mathbf{y}_l\|_2^2\right)^{-1}}. \tag{2}$$

The objective minimizes the symmetrized KL divergence

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}, \qquad p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}. \tag{3}$$

Heavy tails in q_{ij} alleviate the "crowding problem" by allowing distant points to remain apart in the embedding.

2.3 Optimization and Practical Considerations

Gradient descent with momentum and early exaggeration optimizes C. Early exaggeration temporarily multiplies p_{ij} to tighten clusters before relaxing them. Proper perplexity selection (typically 5–50) balances local vs. global structure. Multiple random restarts and learning rate tuning help avoid poor local minima.

3 Applications and Tips

- **Exploratory visualization**: reveal group structure in embeddings of images, text, or single-cell RNA-seq data.
- **Quality assessment**: compare preprocessing pipelines or feature encoders by inspecting t-SNE maps.
- **Prototype selection**: pick representative samples from clusters for labeling or manual review.
- **Best practices**: scale inputs, run with several perplexities, annotate clusters with metadata, and avoid overinterpreting global distances.

4 Python Practice

The script `gen_t_sne_figures.py` standardizes synthetic or real datasets, runs t-SNE at multiple perplexities, and saves the resulting embeddings and a diagnostic curve showing how KL divergence varies with perplexity.

Listing 1: Excerpt from $\text{gen}_{ts}ne_{f}igures.py$

```
from sklearn.manifold import TSNE

perplexities = [10, 30, 50]
embeddings = {}
for perp in perplexities:
    tsne = TSNE(n_components=2, perplexity=perp, learning_rate='auto'
        ,
                init='pca', random_state=42, n_iter=2000)
    embeddings[perp] = tsne.fit_transform(points)
    kl_divergences.append(tsne.kl_divergence_)
```

5 Result

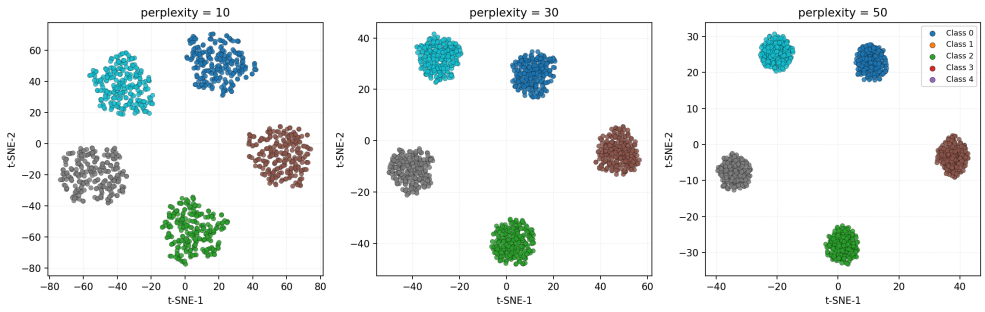

Figure 1: t-SNE embeddings at selected perplexities with coloring by class label

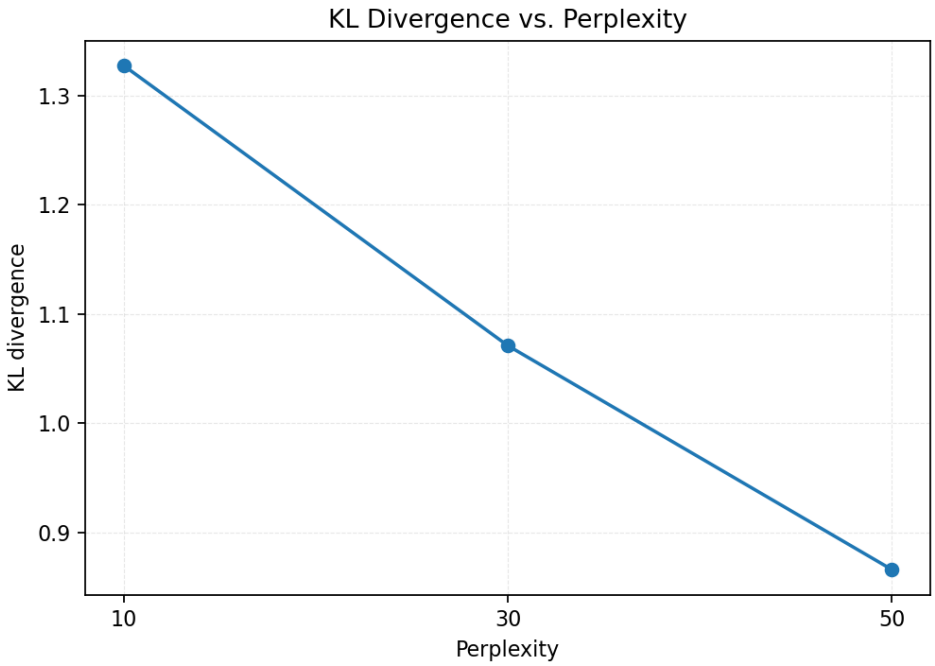

Figure 2: KL divergence versus perplexity illustrating stability ranges

6 Summary

t-SNE captures local neighborhood structure via probabilistic similarity matching and visualizes high-dimensional data with minimal crowding artifacts. Careful tuning of perplexity, learning rate, and iterations yields stable maps that guide exploratory analysis. The example demonstrates how to compare embeddings across perplexities and interpret KL divergence diagnostics.