

# ActUp: Analyzing and Consolidating tSNE & UMAP

**Andrew Draganov<sup>1</sup>, Jakob Jørgensen<sup>1</sup>, Katrine Scheel<sup>1</sup>, Davide Mottin<sup>1</sup>, Ira Assent<sup>1</sup>, Tyrus Berry<sup>2</sup>, Cigdem Aslay<sup>1</sup>**

<sup>1</sup>Aarhus University

<sup>2</sup>George Mason University

{draganovandrew, jakobrj, scheel, davide, ira, cigdem}@cs.au.dk, tberry@gmu.edu

## Abstract

tSNE and UMAP are popular dimensionality reduction algorithms due to their speed and interpretable low-dimensional embeddings. Despite their popularity, however, little work has been done to study their full span of differences. We theoretically and experimentally evaluate the space of parameters in both tSNE and UMAP and observe that a single one – the normalization – is responsible for switching between them. This, in turn, implies that a majority of the algorithmic differences can be toggled without affecting the embeddings. We discuss the implications this has on several theoretic claims behind UMAP, as well as how to reconcile them with existing tSNE interpretations.

Based on our analysis, we provide a method (GDR) that combines previously incompatible techniques from tSNE and UMAP and can replicate the results of either algorithm. This allows our method to incorporate further improvements, such as an acceleration that obtains either method’s outputs faster than UMAP. We release improved versions of tSNE, UMAP, and GDR that are fully plug-and-play with the traditional libraries.

## 1 Introduction

Dimensionality Reduction (DR) algorithms are invaluable for qualitatively inspecting high-dimensional data and are widely used across scientific disciplines. Broadly speaking, these algorithms transform a high-dimensional input into a faithful lower-dimensional embedding. This embedding aims to preserve similarities among the points, where similarity is often measured by distances in the corresponding spaces.

tSNE [Van der Maaten and Hinton, 2008] [Van Der Maaten, 2014] and UMAP [McInnes *et al.*, 2018] are two widely popular DR algorithms due to their efficiency and interpretable embeddings. Both algorithms establish analogous similarity measures, share comparable loss functions, and find an embedding through gradient descent. Despite these similarities, tSNE and UMAP have several key differences. First, although both methods obtain similar results, UMAP prefers large inter-cluster distances while tSNE leans towards

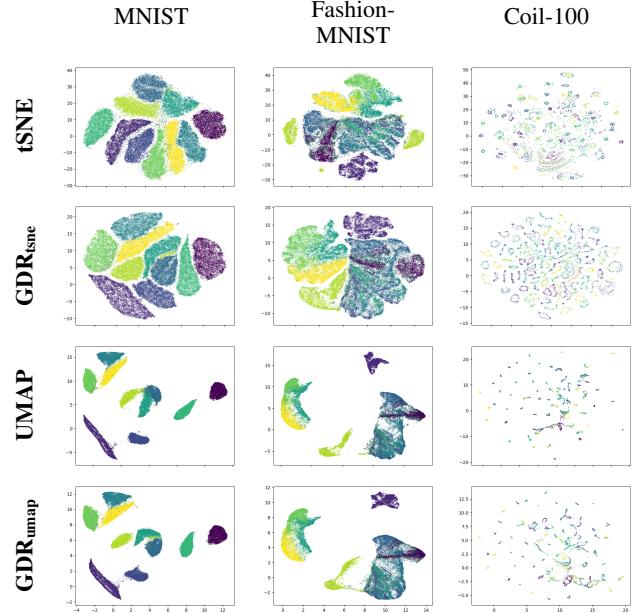


Figure 1: A single method (GDR) can recreate tSNE and UMAP outputs just by changing the normalization.

large intra-cluster distances. Second, UMAP runs significantly faster as it performs efficient sampling during gradient descent. While attempts have been made to study the gaps between the algorithms [Kobak and Linderman, 2021], [Bohm *et al.*, 2020], [Damrich and Hamprecht, 2021], there has not yet been a comprehensive analysis of their methodologies nor a method that can obtain both tSNE and UMAP embeddings at UMAP speeds.

We believe that this is partly due to their radically different presentations. While tSNE takes a computational angle, UMAP originates from category theory and topology. Despite this, many algorithmic choices in UMAP and tSNE are presented without theoretical justification, making it difficult to know which algorithmic components are necessary.

In this paper, we make the surprising discovery that the differences in *both* the embedding structure *and* computational complexity between tSNE and UMAP can be resolved via a single algorithmic choice – the normalization factor. We come to this conclusion by deriving both algorithms from

first principles and theoretically showing the effect that the normalization has on the gradient structure. We supplement this by identifying every implementation and hyperparameter difference between the two methods and implementing tSNE and UMAP in a common library. Thus, we study the effect that each choice has on the embeddings and show both quantitatively and qualitatively that, other than the normalization of the pairwise similarity matrices, none of these parameters significantly affect the outputs.

Based on this analysis, we introduce the necessary changes to the UMAP algorithm such that it can produce tSNE embeddings as well. We refer to this algorithm as Gradient Dimensionality Reduction (GDR) to emphasize that it is consistent with the presentations of *both* tSNE and UMAP. We experimentally validate that GDR can simulate both methods through a thorough quantitative and qualitative evaluation across many datasets and settings. Lastly, our analysis provides insights for further speed improvements and allows GDR to perform gradient descent faster than the standard implementation of UMAP.

In summary, our contributions are as follows:

1. We perform the first comprehensive analysis of the differences between tSNE and UMAP, showing the effect of each algorithmic choice on the embeddings.
2. We theoretically and experimentally show that changing the normalization is a sufficient condition for switching between the two methods.
3. We release simple, plug-and-play implementations of GDR, tSNE and UMAP that can toggle all of the identified hyperparameters. Furthermore, GDR obtains embeddings for both algorithms faster than UMAP.

## 2 Related Work

When discussing tSNE we are referring to [Van Der Maaten, 2014] which established the nearest neighbor and sampling improvements and is generally accepted as the standard tSNE method. A popular subsequent development was presented in [Linderman *et al.*, 2019], wherein Fast Fourier Transforms were used to accelerate the comparisons between points. Another approach is LargeVis [Tang *et al.*, 2016], which modifies the embedding functions to satisfy a graph-based Bernoulli probabilistic model of the low-dimensional dataset. As the more recent algorithm, UMAP has not had as many variations yet. One promising direction, however, has extended UMAP’s second step as a parametric optimization on neural network weights [Sainburg *et al.*, 2020].

Many of these approaches utilize the same optimization structure where they iteratively attract and repel points. While most perform their attractions along nearest neighbors in the high-dimensional space, the repulsions are the slowest operation and each method approaches them differently. tSNE samples repulsions by utilizing Barnes-Hut (BH) trees to sum the forces over distant points. The work in [Linderman *et al.*, 2019] instead calculates repulsive forces with respect to specifically chosen interpolation points, cutting down on the  $\mathcal{O}(n \log n)$  BH tree computations. UMAP and LargeVis, on the other hand, simplify the repulsion sampling by only calculating the gradient with respect to a constant number of

points. These repulsion techniques are, on their face, incompatible with one another, i.e., several modifications have to be made to each algorithm before one can interchange the repulsive force calculations.

There is a growing amount of work that compares tSNE and UMAP through a more theoretical analysis [Damrich and Hamprecht, 2021][Bohm *et al.*, 2020][Damrich *et al.*, 2022][Kobak and Linderman, 2021]. [Damrich and Hamprecht, 2021] find that UMAP’s algorithm does not optimize the presented loss and provide its effective loss function. Similarly [Bohm *et al.*, 2020] analyze tSNE and UMAP through their attractive and repulsive forces, discovering that UMAP diverges when using  $\mathcal{O}(n)$  repulsions per epoch. We expand on the aforementioned findings by showing that the forces are solely determined by the choice of normalization, giving a practical treatment to the proposed ideas. Lastly, [Damrich *et al.*, 2022] provides the interesting realization that tSNE and UMAP can both be described through contrastive learning approaches. Our work differs from theirs in that we analyze the full space of parameters in the algorithms and distill the difference to a single factor, allowing us to connect the algorithms without the added layers of contrastive learning theory. Lastly, the authors in [Kobak and Linderman, 2021] make the argument that tSNE can perform UMAP’s manifold learning if given UMAP’s initialization. Namely, tSNE randomly initializes the low dimensional embedding whereas UMAP starts from a Laplacian Eigenmap [Belkin and Niyogi, 2003] projection. While this may help tSNE preserve the local  $k$ NN structure of the manifold, it is not true of the macro-level distribution of the embeddings. Lastly, [Wang *et al.*, 2021] discusses the role that the loss function has on the resulting embedding structure. This is in line with our results, as we show that the normalization’s effect on the loss function is fundamental in the output differences between tSNE and UMAP.

## 3 Comparison of tSNE and UMAP

We begin by formally introducing the tSNE and UMAP algorithms. Let  $X \in \mathbb{R}^{n \times D}$  be a high dimensional dataset of  $n$  points and let  $Y \in \mathbb{R}^{n \times d}$  be a previously initialized set of  $n$  points in lower-dimensional space such that  $d < D$ . Our aim is to define similarity measures between the points in each space and then find the embedding  $Y$  such that the pairwise similarities in  $Y$  match those in  $X$ .

To do this, both algorithms define high- and low-dimensional non-linear functions  $p : X \times X \rightarrow [0, 1]$  and  $q : Y \times Y \rightarrow [0, 1]$ . These form pairwise similarity matrices  $P(X), Q(Y) \in \mathbb{R}^{n \times n}$ , where the  $i, j$ -th matrix entry represents the similarity between points  $i$  and  $j$ . Formally,

$$p_{j|i}^{tsne}(x_i, x_j) = \frac{\exp(-d(x_i, x_j)^2 / 2\sigma_i^2)}{\sum_{k \neq l} \exp(-d(x_k, x_l)^2 / 2\sigma_k^2)} \quad (1)$$

$$q_{ij}^{tsne}(y_i, y_j) = \frac{(1 + \|y_i - y_j\|_2^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|_2^2)^{-1}} \quad (2)$$

$$\begin{aligned} p_{j|i}^{umap}(x_i, x_j) &= \exp((-d(x_i, x_j)^2 + \rho_i) / \tau_i) \\ q_{ij}^{umap}(y_i, y_j) &= (1 + a(\|y_i - y_j\|_2^2)^b)^{-1}, \end{aligned} \quad (2)$$

where  $d(x_i, x_j)$  is the high-dimensional distance function,  $\sigma$  and  $\tau$  are point-specific variance scalars<sup>1</sup>,  $\rho_i = \min_{l \neq i} d(x_i, x_l)$ , and  $a$  and  $b$  are constants. Note that the tSNE denominators in Equation 1 are the sums of all the numerators. We thus refer to tSNE’s similarity functions as being *normalized* while UMAP’s are *unnormalized*.

The high-dimensional  $p$  values are defined with respect to the point in question and are subsequently symmetrized. WLOG, let  $p_{ij} = S(p_{j|i}, p_{i|j})$  for some symmetrization function  $S$ . Going forward, we write  $p_{ij}$  and  $q_{ij}$  without the superscripts when the normalization setting is clear from the context.

Given these pairwise similarities in the high- and low-dimensional spaces, tSNE and UMAP attempt to find the embedding  $Y$  such that  $Q(Y)$  is closest to  $P(X)$ . Since both similarity measures carry a probabilistic interpretation, we find an embedding by minimizing the KL divergence  $KL(P||Q)$ . This gives us:

$$\mathcal{L}_{tsne} = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (3)$$

$$\mathcal{L}_{umap} = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} + (1 - p_{ij}) \log \frac{1 - p_{ij}}{1 - q_{ij}} \quad (4)$$

In essence, tSNE minimizes the KL divergence of the entire pairwise similarity matrix since its  $P$  and  $Q$  matrices sum to 1. UMAP instead defines Bernoulli probability distributions  $\{p_{ij}, 1 - p_{ij}\}, \{q_{ij}, 1 - q_{ij}\}$  and sums the KL divergences between the  $n^2$  pairwise probability distributions<sup>2</sup>.

### 3.1 Gradient Calculations

We now describe and analyze the gradient descent approaches in tSNE and UMAP. First, notice that the gradients of each algorithm change substantially due to the differing normalizations. In tSNE, the gradient can be written as an attractive  $\mathcal{A}_i^{tsne}$  and a repulsive  $\mathcal{R}_i^{tsne}$  force acting on point  $y_i$  with

$$\begin{aligned} \frac{\partial \mathcal{L}_{tsne}}{\partial y_i} &= -4Z \left[ \sum_{j, j \neq i} p_{ij} q_{ij} (y_i - y_j) - \sum_{k, k \neq i} q_{ik}^2 (y_i - y_k) \right] \\ &= 4Z(\mathcal{A}_i^{tsne} + \mathcal{R}_i^{tsne}) \end{aligned} \quad (5)$$

where  $Z$  is the normalization term in  $q_{ij}^{tsne}$ . On the other hand, UMAP’s attractions and repulsions<sup>3</sup> are presented as [McInnes *et al.*, 2018]

$$\mathcal{A}_i^{umap} = \sum_{j, j \neq i} \frac{-2ab \|y_i - y_j\|_2^{2(b-1)}}{1 + \|y_i - y_j\|_2^2} p_{ij} (y_i - y_j) \quad (6)$$

$$\mathcal{R}_i^{umap} = \sum_{k, k \neq i} \frac{2b}{\varepsilon + \|y_i - y_k\|_2^2} q_{ik} (1 - p_{ik}) (y_i - y_k). \quad (7)$$

<sup>1</sup>In practice, we can assume that  $2\sigma_i^2$  is functionally equivalent to  $\tau_i$ , as they are both chosen such that the entropy of the resulting distribution is equivalent.

<sup>2</sup>Both tSNE and UMAP set the diagonals of  $P$  and  $Q$  to 0

<sup>3</sup>The  $\varepsilon$  value is only inserted for numerical stability

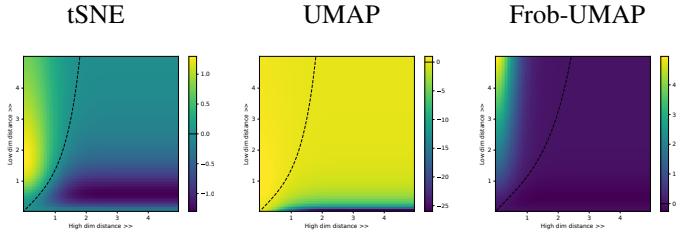


Figure 2: Gradient relationships between high- and low-dimensional distances for tSNE, UMAP, and UMAP under the Frobenius norm. The dotted line represents the locations of magnitude-0 gradients. Higher values correspond to attractions while lower values correspond to repulsions. The left image is a recreation of the original gradient plot in [Van der Maaten and Hinton, 2008].

In the setting where  $a = b = 1$  and  $\varepsilon = 0$ , Equations 6, 7 can be written as<sup>4</sup>

$$\begin{aligned} \mathcal{A}_i^{umap} &= -2 \sum_{j, j \neq i} p_{ij} q_{ij} (y_i - y_j) \\ \mathcal{R}_i^{umap} &= 2 \sum_{k, k \neq i} q_{ik}^2 \frac{1 - p_{ik}}{1 - q_{ik}} (y_i - y_k) \end{aligned} \quad (8)$$

We remind the reader that we are overloading notation –  $p$  and  $q$  are normalized when they are in the tSNE setting and are unnormalized in the UMAP setting.

In practice, tSNE and UMAP optimize their loss functions by iteratively applying these attractive and repulsive forces. It is unnecessary to calculate each such force to effectively estimate the gradient, however, as the  $p_{ij}$  term in both the tSNE and UMAP attractive forces decays exponentially. Based on this observation, both methods establish a nearest neighbor graph in the high-dimensional space, where the edges represent nearest neighbor relationships between  $x_i$  and  $x_j$ . It then suffices to only perform attractions between points  $y_i$  and  $y_j$  if their corresponding  $x_i$  and  $x_j$  are nearest neighbors.

This logic does not transfer to the repulsions, however, as the Student-t distribution has a heavier tail so repulsions must be calculated evenly across the rest of the points. tSNE does this by fitting a Barnes-Hut tree across  $Y$  during *every epoch*. If  $y_k$  and  $y_l$  are both in the same tree leaf then we assume  $q_{ik} = q_{il}$ , allowing us to only calculate  $\mathcal{O}(\log(n))$  similarities. Thus, tSNE estimates all  $n - 1$  repulsions by performing one such estimate for each cell in  $Y$ ’s Barnes-Hut tree. UMAP, on the other hand, simply obtains repulsions by sampling a constant number of points uniformly and only applying those repulsions. These repulsion schemas are depicted in Figure 3. Note, tSNE collects all of the gradients before a full momentum gradient descent step whereas UMAP moves each point immediately upon calculating a force.

There are a few differences between the two algorithms’ gradient descent loops. First, the tSNE learning rate stays constant over training while UMAP’s linearly decreases. Second, tSNE’s gradients are strengthened by adding a “gains” term which scales gradients based on whether they point in

<sup>4</sup>We derive this in section A.2 in the supplementary material

the same direction from epoch to epoch<sup>5</sup>. We refer to these two elements as *gradient amplification*.

Note that UMAP’s repulsive force has a  $1 - p_{ik}$  term that is unavailable at runtime, as  $x_i$  and  $x_k$  may not have been nearest neighbors. In practice, UMAP estimates these  $1 - p_{ik}$  terms by using the available  $p_{ij}$  values<sup>6</sup>. We also note that UMAP does not explicitly multiply by  $p_{ij}$  and  $1 - p_{ik}$ . Instead, it samples the forces proportionally to these scalars. For example, if  $p_{ij} = 0.1$  then we apply that force *without the  $p_{ij}$  multiplier* once every ten epochs. We refer to this as *scalar sampling*.

### 3.2 The Choice of Normalization

We now present a summary of our theoretical results before providing their formal statements. As was shown in [Bohm *et al.*, 2020], the ratio between attractive and repulsive magnitudes determines the structure of the resulting embedding. Given this context, Theorem 1 shows that the normalization directly changes the ratio of attraction/repulsion magnitudes, inducing the difference between tSNE and UMAP embeddings. Thus, we can toggle the normalization to alternate between their outputs. Furthermore, Theorem 2 shows that the attraction/repulsion ratio in the normalized setting is *independent of* the number of repulsive samples collected. This second point allows us to accelerate tSNE to UMAP speeds without impacting embedding quality by simply removing the dependency on Barnes-Hut trees and calculating 1 per-point repulsion as in UMAP. We now provide the necessary definitions for the theorems.

Assume that the  $p_{ij}$  terms are given. We now consider the dataset  $Y$  probabilistically by defining a set of random variables  $v_{ij} = y_i - y_j$  and assume that all  $\mathcal{O}(n^2)$   $v_{ij}$  vectors are i.i.d. around a non-zero mean. Let  $r_{ij} = (1 + |v_{ij}|^2)^{-1}$  and define  $Z = \sum_{i,j} r_{ij}$  as the sum over  $n^2$  pairs of points and  $\tilde{Z} = \sum_{i,j} r_{ij}$  as the sum over  $n$  pairs of points. Then applying  $n$  per-point repulsions gives us the force acting on point  $y_i$  of  $\mathbb{E}[|\mathcal{R}_{tsne}|] = \mathbb{E}[\sum_j ||(r_{ij}^2/Z^2) \cdot v_{ij}||]$ . We now define an equivalent force term in the setting where we have 1 per-point repulsion:  $\mathbb{E}[|\tilde{\mathcal{R}}_{tsne}|] = \mathbb{E}[||(r_{ij}^2/\tilde{Z}^2) \cdot v_{ij}||]$ . Note that we have a constant number  $c$  of attractive forces acting on each point, giving  $\mathbb{E}[|\mathcal{A}_{tsne}|] = c \cdot p_{ij}^{tsne} \mathbb{E}[||(r_{ij}/Z) \cdot v_{ij}||]$  and  $\mathbb{E}[|\tilde{\mathcal{A}}_{tsne}|] = c \cdot p_{ij}^{tsne} \mathbb{E}[||(r_{ij}/\tilde{Z}) \cdot v_{ij}||]$ .

Thus,  $|\mathcal{A}_{tsne}|$  and  $|\mathcal{R}_{tsne}|$  represent the magnitudes of the forces when we calculate tSNE’s  $\mathcal{O}(n)$  per-point repulsions while  $|\tilde{\mathcal{A}}_{tsne}|$  and  $|\tilde{\mathcal{R}}_{tsne}|$  represent the forces when we have UMAP’s  $\mathcal{O}(1)$  per-point repulsions. Given this, we have the following theorems:

**Theorem 1.** Let  $p_{ij}^{tsne} \sim 1/(cn)$  and  $d(x_i, x_j) > \sqrt{\log(n^2 + 1)\tau}$ . Then  $\frac{\mathbb{E}[|\mathcal{A}_i^{tsne}|]}{\mathbb{E}[|\mathcal{R}_i^{tsne}|]} < \frac{\mathbb{E}[|\tilde{\mathcal{A}}_i^{tsne}|]}{\mathbb{E}[|\tilde{\mathcal{R}}_i^{tsne}|]}$ .

<sup>5</sup>This term has not been mentioned in the literature but is present in common tSNE implementations.

<sup>6</sup>When possible, we use index  $k$  to represent repulsions and  $j$  to represent attractions to highlight that  $p_{ik}$  is never calculated in UMAP. See Section A.6 in the supplementary material for details.

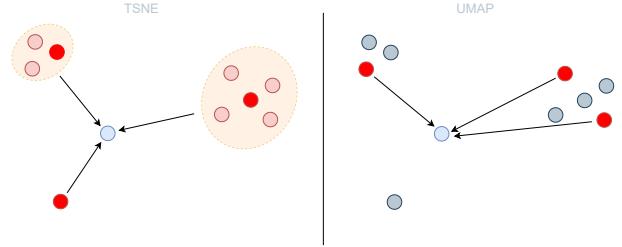


Figure 3: Visualization of the repulsive forces in tSNE (left) and UMAP (right). tSNE calculates the repulsion for representative points and uses this as a proxy for nearby points, giving  $\mathcal{O}(n)$  total repulsions acting on each point. UMAP calculates the repulsion to a pre-defined number of points and ignores the others, giving  $\mathcal{O}(1)$  per-point repulsions. Bright red points are those for which the gradient is calculated; arrows are the direction of repulsion.

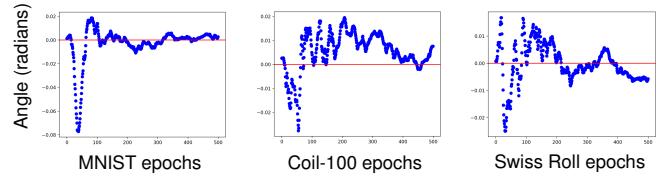


Figure 4: Average angle in radians between repulsive forces calculated with  $\mathcal{O}(1)$  and  $\mathcal{O}(n)$  repulsions. The red line is at 0 radians.

$$\textbf{Theorem 2. } \frac{\mathbb{E}[|\mathcal{A}_i^{tsne}|]}{\mathbb{E}[|\mathcal{R}_i^{tsne}|]} = \frac{\mathbb{E}[|\tilde{\mathcal{A}}_i^{tsne}|]}{\mathbb{E}[|\tilde{\mathcal{R}}_i^{tsne}|]}$$

The proofs are given in Sections A.3 and A.4 of the supplementary material. We point out that  $p_{ij}^{tsne}$  is normalized over the sum of all  $cn$  attractions that are sampled, giving us the estimate  $p_{ij}^{tsne} \sim 1/(cn)$ . Theorem 1’s result is visualized in the gradient plots in Figure 2. There we see that, for non-negligible values of  $d(x_i, x_j)$ , the UMAP repulsions can be orders of magnitude larger than the corresponding tSNE ones, even when accounting for the magnitude of the attractions. Furthermore, Section 5 evidences that toggling the normalization is sufficient to switch between the algorithms’ embeddings and that no other hyperparameter accounts for the difference in inter-cluster distances between tSNE and UMAP.

### 4 Unifying tSNE and UMAP

This leads us to GDR– a modification to UMAP that can recreate both tSNE and UMAP embeddings at UMAP speeds. We choose the general name Gradient Dimensionality Reduction to imply that it is *both* UMAP and tSNE.

Our algorithm follows the UMAP optimization procedure except that we (1) replace the *scalar sampling* by iteratively processing attractions/repulsions and (2) apply the gradients after having collected all of them, rather than immediately upon processing each one. The first change accommodates the gradients under normalization since the normalized repulsive forces do not have the  $1 - p_{ik}$  term to which UMAP samples proportionally. The second change allows for performing momentum gradient descent for faster convergence

<b>Initialization</b>	<b>Distance function</b>	<b>Symmetrization</b>	<b>Sym Attraction</b>	<b>Scalars</b>
<i>Y initialization</i>	<i>High-dim distances calculation</i>	<i>Setting <math>p_{ij} = p_{ji}</math></i>	<i>Attraction(<math>y_i, y_j</math>) applied to both</i>	<i>Values for <math>a</math> and <math>b</math></i>
tSNE	Random	$d(x_i, x_j)$	$(p_{i j} + p_{j i})/2$	No
UMAP	Lapl. Eigenmap	$d(x_i, x_j) - \min_k d(x_i, x_k)$	$p_{i j} + p_{j i} - p_{i j}p_{j i}$	Yes Grid search

Table 1: List of differences between hyperparameters of tSNE and UMAP. These are analyzed in Figures 3 and 4.

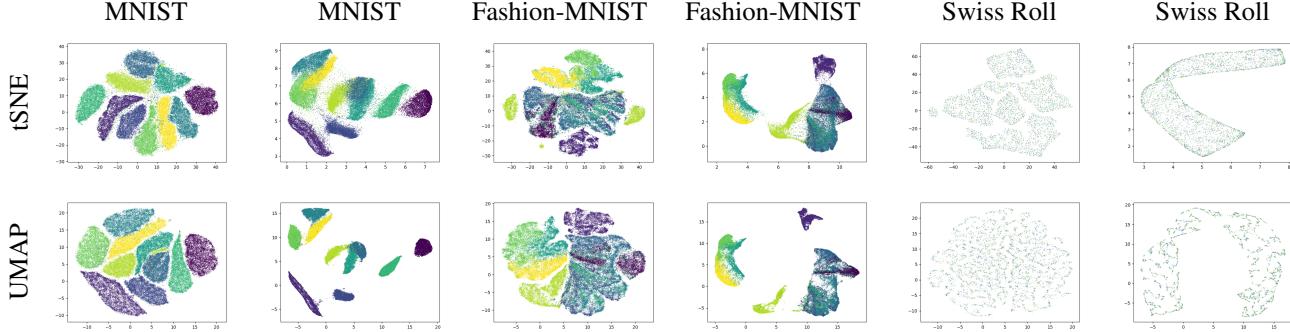


Table 2: Effect of changing the normalization for the original tSNE and UMAP algorithms on the MNIST, Fashion-MNIST, and Swiss Roll datasets. Each dataset is shown with normalization followed by no normalization. We use Laplacian Eigenmap initializations for consistent orientation. The normalized UMAP plots were made with the changes described in section 5.2.

in the normalized setting.

Since we follow the UMAP optimization procedure, GDR defaults to producing UMAP embeddings. In the case of replicating tSNE, we simply normalize the  $P$  and  $Q$  matrices and scale the learning rate. Although we only collect  $\mathcal{O}(1)$  attractions and repulsions for each point, their magnitudes are balanced due to Theorems 1 and 2. We refer to GDR as  $\text{GDR}_{\text{umap}}$  if it is in the unnormalized setting and as  $\text{GDR}_{\text{tsne}}$  if it is in the normalized setting. We note that changing the normalization necessitates gradient amplification.

By allowing GDR to toggle the normalization, we are free to choose the simplest options across the other parameters. GDR therefore defaults to tSNE’s asymmetric attraction and  $a$  and  $b$  scalars along with UMAP’s distance-metric, initialization, nearest neighbors, and  $p_{ij}$  symmetrization.

The supplementary material provides some further information on the flexibility of GDR (A.1), such as an accelerated version of the algorithm where we modify the gradient formulation such that it is quicker to optimize. This change induces a consistent  $2\times$  speedup of GDR over UMAP. Despite differing from the true KL divergence gradient, we find that the resulting embeddings are comparable. Our repository also provides a CUDA kernel that calculates  $\text{GDR}_{\text{umap}}$  and  $\text{GDR}_{\text{tsne}}$  embeddings in a distributed manner on a GPU.

## 4.1 Theoretical Considerations

UMAP’s theoretical framework identifies the existence of a locally-connected manifold in the high-dimensional space under the UMAP pseudo-distance metric  $\tilde{d}$ . This pseudo-distance metric is defined such that the distance from point  $x_j$  to  $x_i$  is equal to  $\tilde{d}(x_i, x_j) = d(x_i, x_j) - \min_{l \neq i} d(x_i, x_l)$ . Despite this being a key element of the UMAP foundation, we find that substituting the Euclidean distance for the pseudo-

distance metric seems to have no effect on the embeddings, as seen in Tables 3 and 4. It is possible that the algorithm’s reliance on highly non-convex gradient descent deviates enough from the theoretical discussion that the pseudo-distance metric loses its applicability. It may also be the case that this pseudo-distance metric, while insightful from a theoretical perspective, is not a necessary calculation in order to achieve the final embeddings.

Furthermore, many of the other differences between tSNE and UMAP are not motivated by the theoretical foundation of either algorithm. The gradient descent methodology is entirely heuristic, so any differences therein do not impact the theory. This applies to the repulsion and attraction sampling and gradient descent methods. Moreover, the high-dimensional symmetrization function, embedding initialization, symmetric attraction, and  $a, b$  scalars can all be switched to their alternative options without impacting either method’s consistency within its theoretical presentation. Thus, each of these heuristics can be toggled without impacting the embedding’s interpretation, as most of them do not interfere with the theory and none affect the output.

We also question whether the choice of normalization is necessitated by either algorithm’s presentation. tSNE, for example, treats the normalization of  $P$  and  $Q$  as an assumption and provides no further justification. In the case of UMAP, it appears that the normalization does not break the assumptions of the original paper [McInnes *et al.*, 2018, Sec. 2,3]. We therefore posit that the interpretation of UMAP as finding the best fit to the high-dimensional data manifold extends to tSNE as well, as long as tSNE’s gradients are calculated under the pseudo-distance metric in the high-dimensional space. We additionally theorize that each method can be paired with either normalization without contradicting the foundations

	Fashion MNIST	Coil 100	Single Cell	Cifar-10
kNN Acc.	UMAP 78.0; 0.5	80.8; 3.3	43.4; 1.9	24.2; 1.1
V-score	GDR <sub>umap</sub> 77.3; 0.7	77.4; 3.4	42.8; 2.2	23.8; 1.1
	tSNE 80.1; 0.7	63.2; 4.2	43.3; 1.9	28.7; 2.5
	GDR <sub>tsne</sub> 78.6; 0.6	77.2; 4.4	44.8; 1.4	25.6; 1.1
	UMAP 60.3; 1.4	89.2; 0.9	60.6; 1.3	7.6; 0.4
	GDR <sub>umap</sub> 61.7; 0.8	91.0; 0.6	60.1; 1.6	8.1; 0.6
	tSNE 54.2; 4.1	82.9; 1.8	59.7; 1.1	8.5; 0.3
	GDR <sub>tsne</sub> 51.7; 4.7	85.7; 2.6	60.5; 0.8	8.0; 3.7

Table 3: Row means and std. deviations for kNN-accuracy and V-score on Fashion MNIST, Coil-100, Single-Cell, and Cifar-10 datasets. For example, the cell [Fashion-MNIST, kNN accuracy, tSNE] implies that the mean kNN accuracy across the hyperparameters in Table 1 was 80.1 for tSNE on the Fashion-MNIST dataset.

laid out in its paper.

We evidence the fact that tSNE can preserve manifold structure at least as well as UMAP in Table 2, where Barnes-Hut tSNE without normalization cleanly maintains the structure of the Swiss Roll dataset. We further discuss these manifold learning claims in the supplementary material (A.5).

For all of these reasons, we make the claim that tSNE and UMAP are computationally consistent with one another. That is, we conjecture that, up to minor changes, one could have presented UMAP’s theoretical foundation and implemented it with the tSNE algorithm or vice-versa.

## 4.2 Frobenius Norm for UMAP

Finally, even some of the standard algorithmic choices can be modified without significantly impacting the embeddings. For example, UMAP and tSNE both optimize the KL divergence, but we see no reason that the Frobenius norm cannot be substituted in its place. Interestingly, the embeddings in Figure 8 in the supplementary material show that optimizing the Frobenius norm in the unnormalized setting produces outputs that are indistinguishable from the ones obtained by minimizing the KL-divergence. To provide a possible indication as to why this occurs, Figure 2 shows that the zero-gradient areas between the KL divergence and the Frobenius norm strongly overlap, implying that a local minimum under one objective satisfies the other one as well.

We bring this up for two reasons. First, the Frobenius norm is a significantly simpler loss function to optimize than the KL divergence due to its convexity. We hypothesize that there must be simple algorithmic improvements that can exploit this property. Further detail is given in Section A.7 in the supplementary material. Second, it is interesting to consider that even fundamental assumptions such as the objective function can be changed without significantly affecting the embeddings across datasets.

## 5 Results

**Metrics.** There is no optimal way to compare embeddings – an analysis at the point-level loses global information while

studying macro-structures loses local information. To account for this, we employ separate metrics to study the embeddings at the micro- and macro-scales. Specifically, we use the kNN accuracy to analyze preservation of local neighborhoods as established in [Van Der Maaten *et al.*, 2009] and the V-measure [Rosenberg and Hirschberg, 2007] to study the embedding’s global cluster structures<sup>7</sup>.

## 5.1 Hyperparameter Effects

We first show that a majority of the differences between tSNE and UMAP do not significantly affect the embeddings. Specifically, Table 4 shows that we can vary the hyperparameters in Table 1 with negligible change to the embeddings of any discussed algorithm. Equivalent results on other datasets can be found in Tables 8 and 9 in the supplementary material. Furthermore, Table 3 provides quantitative evidence that the hyperparameters do not affect the embeddings across datasets; similarly, Table 9 in the supplementary material confirms this finding across algorithms.

Looking at Table 4, the initialization and the symmetric attraction induce the largest variation in the embeddings. For the initialization, the relative positions of clusters change but the relevant inter-cluster relationships remain consistent<sup>8</sup>. Enabling symmetric attraction attracts  $y_j$  to  $y_i$  when we attract  $y_i$  to  $y_j$ . Thus, switching from asymmetric to symmetric attraction functionally scales the attractive force by 2. This leads to tighter tSNE clusters that would otherwise be evenly spread out across the embedding, but does not affect UMAP significantly. We thus choose asymmetric attraction for GDR as it better recreates tSNE embeddings.

We show the effect of *single* hyperparameter changes for combinatorial reasons. However, we see no significant difference between changing one hyperparameter or any number of them. We also eschew including hyperparameters that have no effect on the embeddings and are the least interesting. These include the exact vs. approximate nearest neighbors, gradient clipping, and the number of epochs.

## 5.2 Effect of Normalization

Although Theorem 2 shows that we can take fewer repulsive samples without affecting the repulsion’s magnitude, we must also verify that the angle of the repulsive force is preserved as well. Towards this end, we plot the average angle between the tSNE Barnes-Hut repulsions and the UMAP sampled repulsions in Figure 4. We see that, across datasets, the direction of the repulsion remains consistent throughout the optimization process. Thus, since both the magnitude and the direction are robust to the number of samples taken, we conclude that one can obtain tSNE embeddings with  $\mathcal{O}(1)$  per-point repulsions.

We now show that toggling the normalization allows tSNE to simulate UMAP embeddings and vice versa. Table 2 shows exactly this. First note that tSNE in the unnormalized setting has significantly more separation between clusters in a manner similar to UMAP. The representations are fuzzier than the

<sup>7</sup>We provide formalization of these metrics in the supplementary material (A.8)

<sup>8</sup>As such, we employ the Laplacian Eigenmap initialization on small datasets ( $< 100K$ ) due to its predictable output and the random initialization on large datasets ( $> 100K$ ) to avoid slowdowns.

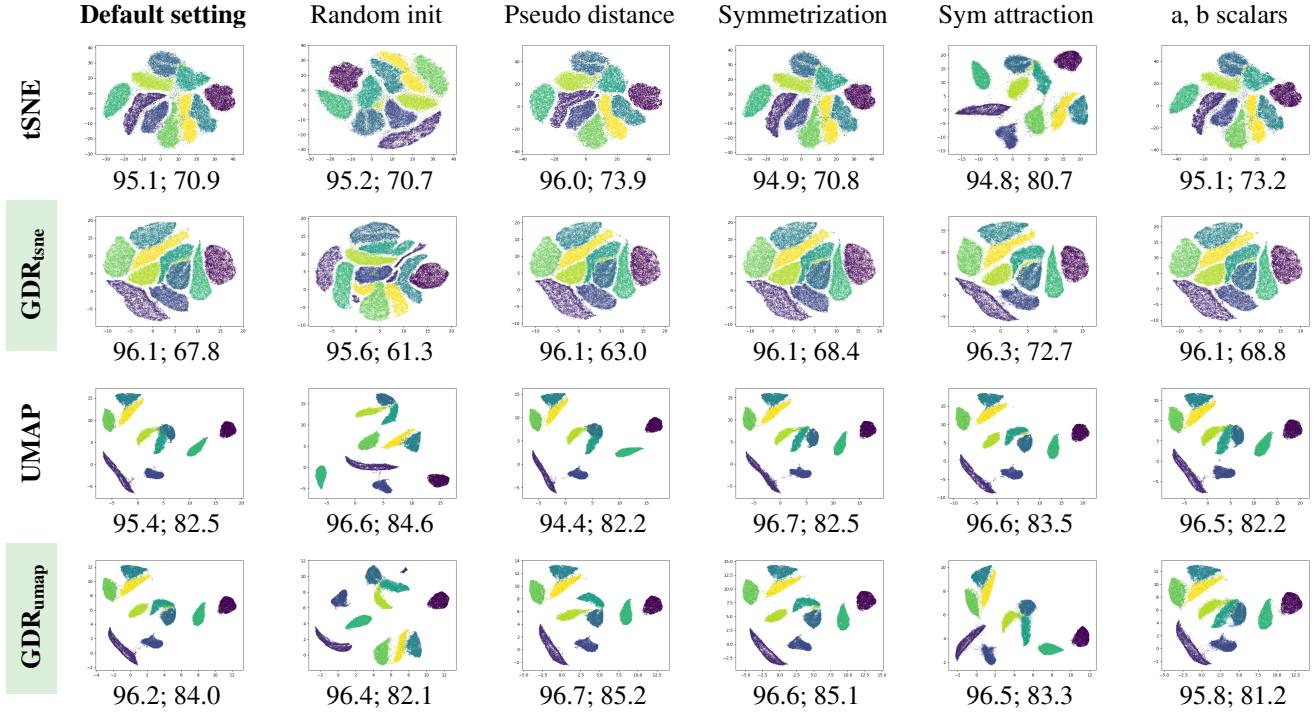


Table 4: Effect of the algorithm settings from Table 1 on the MNIST dataset. Each parameter is changed from its default to its alternative setting; e.g., the random init column implies that tSNE was initialized with Laplacian Eigenmaps while UMAP and GDR were initialized randomly. Below each image the KNN-accuracy and K-Means V-score show unchanged performance.

UMAP ones as we are still estimating  $\mathcal{O}(n)$  repulsions, causing the embedding to fall closer to the mean of the multi-modal datasets. To account for the  $n \times$  more repulsions, we scale each repulsion by  $1/n$  for the sake of convergence. This is a different effect than normalizing by  $\sum p_{ij}$  as we are not affecting the attraction/repulsion ratio in Theorem 1.

The analysis is slightly more involved in the case of UMAP. Recall that the UMAP algorithm approximates the  $p_{ij}$  and  $1 - p_{ik}$  gradient scalars by sampling the attractions and repulsions proportionally to  $p_{ij}$  and  $1 - p_{ik}$ , which we referred to as *scalar sampling*. However, the gradients in the normalized setting (Equation 5) lose the  $1 - p_{ik}$  scalar on repulsions. The UMAP optimization schema, then, imposes an unnecessary weight on the repulsions in the normalized setting as the repulsions are still sampled according to the no-longer-necessary  $1 - p_{ik}$  scalar. Accounting for this requires dividing the repulsive forces by  $1 - p_{ik}$ , but this (with the momentum gradient descent and stronger learning rate) leads to a highly unstable training regime. We refer the reader to Figure 7 in the supplementary material for details.

This implies that stabilizing UMAP in the normalized setting requires removing the sampling and instead directly multiplying by  $p_{ij}$  and  $1 - p_{ik}$ . Indeed, this is exactly what we do in GDR. Under this change, GDR<sub>umap</sub> and GDR<sub>tSNE</sub> obtain effectively identical embeddings to the default UMAP and tSNE ones. This is confirmed in the kNN accuracy and K-means V-score metrics in Table 3.

### 5.3 Time Efficiency

We lastly discuss the speeds of UMAP, tSNE, GDR, and our accelerated version of GDR in section A.1 of the supplementary material due to space concerns. Our implementations of UMAP and GDR perform gradient descent an order of magnitude faster than the standard UMAP library, implying a corresponding speedup over tSNE. We also provide an acceleration by doing GDR *with* scalar sampling that provides a further  $2\times$  speedup. Despite the fact that this imposes a slight modification onto the effective gradients, we show that this is qualitatively insignificant in the resulting embeddings.

## 6 Conclusion & Future Work

We discussed the set of differences between tSNE and UMAP and identified that only the normalization significantly impacts the outputs. This provides a clear unification of tSNE and UMAP that is both theoretically simple and easy to implement. Beyond this, our analysis has uncovered multiple misunderstandings regarding UMAP and tSNE while hopefully also clarifying how these methods work.

We raised several questions regarding the theory of gradient-based DR algorithms. Is there a setting in which the UMAP pseudo-distance changes the embeddings? Does the KL divergence induce a better optimization criterium than the Frobenius norm? Is it true that UMAP’s framework can accommodate tSNE’s normalization? We hope that we have facilitated future research into the essence of these algorithms through identifying all of their algorithmic components and consolidating them in a simple-to-use codebase.

## References

- [Belkin and Niyogi, 2003] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.
- [Bohm *et al.*, 2020] Jan Niklas Bohm, Philipp Berens, and Dmitry Kobak. A unifying perspective on neighbor embeddings along the attraction-repulsion spectrum. *arXiv preprint arXiv:2007.08902*, 2020.
- [Damrich and Hamprecht, 2021] Sebastian Damrich and Fred A Hamprecht. On umap’s true loss function. *Advances in Neural Information Processing Systems*, 34, 2021.
- [Damrich *et al.*, 2022] Sebastian Damrich, Jan Niklas Böhm, Fred A Hamprecht, and Dmitry Kobak. Contrastive learning unifies t-sne and umap. *arXiv preprint arXiv:2206.01816*, 2022.
- [Deng, 2012] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.
- [Dong *et al.*, 2011] Wei Dong, Charikar Moses, and Kai Li. Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th international conference on World wide web*, pages 577–586, 2011.
- [Hull, 1994] J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.
- [Kobak and Linderman, 2021] Dmitry Kobak and George C Linderman. Initialization is critical for preserving global data structure in both t-sne and umap. *Nature biotechnology*, 39(2):156–157, 2021.
- [Krizhevsky, 2009] Alex Krizhevsky. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*, 2009.
- [Linderman *et al.*, 2019] George C Linderman, Manas Rachh, Jeremy G Hoskins, Stefan Steinerberger, and Yuval Kluger. Fast interpolation-based t-sne for improved visualization of single-cell rna-seq data. *Nature methods*, 16(3):243–245, 2019.
- [McInnes *et al.*, 2018] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
- [Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [NENE, 1996] SA NENE. Columbia object image library (coil-100). *Technical Report CU-CS-006-96*, 1996.
- [Rosenberg and Hirschberg, 2007] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, pages 410–420, 2007.
- [Sainburg *et al.*, 2020] Tim Sainburg, Leland McInnes, and Timothy Q Gentner. Parametric umap embeddings for representation and semi-supervised learning. *arXiv preprint arXiv:2009.12981*, 2020.
- [Tang *et al.*, 2016] Jian Tang, Jingzhou Liu, Ming Zhang, and Qiaozhu Mei. Visualizing large-scale and high-dimensional data. In *TheWebConf*, pages 287–297, 2016.
- [Tasic *et al.*, 2018] Bosiljka Tasic, Zizhen Yao, Lucas T Graybuck, Kimberly A Smith, Thuc Nghi Nguyen, Darren Bertagnolli, Jeff Goldy, Emma Garren, Michael N Economo, Sarada Viswanathan, et al. Shared and distinct transcriptomic cell types across neocortical areas. *Nature*, 563(7729):72–78, 2018.
- [Van der Maaten and Hinton, 2008] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.
- [Van Der Maaten *et al.*, 2009] Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10(66-71):13, 2009.
- [Van Der Maaten, 2014] Laurens Van Der Maaten. Accelerating t-sne using tree-based algorithms. *JMLR*, 15(1):3221–3245, 2014.
- [Wang *et al.*, 2021] Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *The Journal of Machine Learning Research*, 22(1):9129–9201, 2021.
- [Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.