Statistics and Computing (2023) 33:12
https://doi.org/10.1007/s11222-022-10186-z

ORIGINAL PAPER

Stochastic cluster embedding

Zhirong Yang[1,2] · Yuwei Chen[3] · Denis Sedov[2] · Samuel Kaski[2,7] · Jukka Corander[4,5,6]

Received: 7 October 2021 / Accepted: 20 November 2022

Abstract

Neighbor embedding (NE) aims to preserve pairwise similarities between data items and has been shown to yield an effective principle for data visualization. However, even the best existing NE methods such as stochastic neighbor embedding (SNE) may leave large-scale patterns hidden, for example clusters, despite strong signals being present in the data. To address this, we propose a new cluster visualization method based on the Neighbor Embedding principle. We first present a family of Neighbor Embedding methods that generalizes SNE by using non-normalized Kullback–Leibler divergence with a scale parameter. In this family, much better cluster visualizations often appear with a parameter value different from the one corresponding to SNE. We also develop an efficient software that employs asynchronous stochastic block coordinate descent to optimize the new family of objective functions. Our experimental results demonstrate that the method consistently and substantially improves the visualization of data clusters compared with the state-of-the-art NE approaches. The code of our method is publicly available at https://github.com/rozyangno/sce.

Keywords Clustering · Information divergence · Neighbor embedding · Nonlinear dimensionality reduction · Stochastic optimization · Visualization

1 Introduction

The rapid growth in the amount of data processed by analysts demands more efficient information digestion and communication methods. Data visualization by dimensionality reduction facilitates a viewer to digest information in massive data sets quickly. Therefore, it is increasingly applied as a critical component in scientific research, digital libraries, data mining, financial data analysis, market studies, manufacturing production control, drug discovery, etc.

Stochastic Neighbor Embedding (SNE; Hinton and Roweis 2003) is a widely used nonlinear dimensionality reduction (NLDR) method, which approximately preserves the pairwise probabilities of being neighbors (neighboring probabilities for short) in the input space. In particular, the Student t-Distributed Stochastic Neighbor Embedding (t-SNE; van der Maaten and Hinton 2008) has become one of the most popular nonlinear dimensionality reduction methods for data visualization. The t-SNE method employs a heavy-tailed distribution for the neighboring probabilities in the embedding and minimizes their Kullback–Leibler divergence against precomputed input probabilities.

Discovery of large-scale patterns such as clusters is an important task of NLDR. It is often believed that t-SNE can show clusters for well clusterable data, with a smaller Kullback–Leibler divergence corresponding to a better quality. However, recently we found many counter-examples where t-SNE may not correctly visualize the clusters even if the input neighborhoods are well clusterable (see Sect. 5

✉ Zhirong Yang
zhirong.yang@ntnu.no

Yuwei Chen
yuwei.chen@maanmittauslaitos.fi

Denis Sedov
denis.sedov@aalto.fi

Samuel Kaski
samuel.kaski@aalto.fi

Jukka Corander
jukka.corander@medisin.uio.no

[1] Norwegian University of Science and Technology, Trondheim, Norway

[2] Aalto University, Espoo, Finland

[3] Finnish Geospatial Research Institute, Espoo, Finland

[4] University of Oslo, Oslo, Norway

[5] Wellcome Sanger Institute, Saffron Walden, UK

[6] University of Helsinki, Helsinki, Finland

[7] University of Manchester, Mancherster, UK

Published online: 04 December 2022

for details). Tuning hyperparameters in t-SNE does not help in solving this issue because a better t-SNE learning objective can correspond to a worse cluster embedding. The root cause of the failure is that t-SNE was designed to preserve the neighborhood or local information, which is not necessarily directly connected with finding large-scale patterns such as clusters.

To address the problem, we propose a new method called Stochastic Cluster Embedding (SCE) for better cluster visualizations. We generalize SNE using a family of I-divergences parameterized by a scaling factor s, between the non-normalized similarities in the input and output space. We show that SNE is a special case in the family with s chosen to be the normalizing factor of output similarities. However, through a user study, we find that the best s value for cluster visualization is often different from the one chosen by SNE.

To overcome the t-SNE drawback, SCE employs another choice that mixes the input similarities in calculating s. The scale factor is adaptively adjusted when optimizing the new learning objective, and the data points are thus better grouped. We have also developed an efficient optimization algorithm that employs asynchronous stochastic block coordinate descent. The new algorithm can utilize parallel computing devices such as CPUs or GPUs and is suitable for mega-scale problems with massive data items.

Our new method is tested on six real-world data sets and compared with the state-of-the-art nonlinear dimensionality reduction approaches, including t-SNE (van der Maaten and Hinton 2008). The results show that our method consistently performs better than t-SNE for cluster visualization. In all tested cases where t-SNE fails, SCE can show the clusters clearly and accurately. Among all compared methods, our new method is the only one that gives good cluster visualizations for all tested data sets.

The remainder of the paper is organized as follows. The popular visualization method SNE is briefly reviewed in Sect. 2. In Sect. 3 we give our notations about clustering and cluster visualization. In Sect. 4, we present the new method based on Neighbor Embedding for cluster visualization, including its learning objective function and optimization algorithms. We present the experiment settings and results in Sect. 5 and conclude the paper by presenting possible directions for future research in Sect. 6.

2 Stochastic neighbor embedding

Neighbor Embedding (NE) is a collection of methods that seek vectorial representation of data items according to their neighborhoods or similarities in the original space (Yang et al. 2013, 2014). Given a set of multivariate data points $\{x_1, x_2, \ldots, x_N\}$, where $x_i \in \mathbb{R}^D$, their pairwise similarities are encoded in a nonnegative square matrix p. Neighbor Embedding finds a mapping $x_i \mapsto y_i \in \mathbb{R}^d$ for $i = 1, \ldots, N$ ($d = 2$ or $d = 3$ for visualization) such that the similarities in the mapped space, encoded by q_{ij}, approximate those in p. The approximation is realized by minimizing a certain divergence between p and q. For abbreviation we also write $Y = [y_1, \ldots, y_N]$.

SNE is a family of Neigbhor Embedding methods, which minimizes the Kullback–Leibler divergence $D_{\mathrm{KL}}(P \,||\, Q)$ between the normalized input and output similarities P and Q. The matrix P can be precomputed, for example, $P_{ij} = (P_{i|j} + P_{j|i})/(2N)$ with $P_{i|j} = p_{ij}/\sum_l p_{il}$. In this paper we focus on the matrix-wise normalization[1] of Q using $Q_{ij} = q_{ij}/\sum_{ab} q_{ab}$. In SNE, the output similarities are usually given by a Gaussian kernel $q_{ij} = \exp\left(-\|y_i - y_j\|^2\right)$ or a Student t-kernel $q_{ij} = (1 + \|y_i - y_j\|^2)^{-1}$. SNE equipped with the latter kernel is called Student t-distributed Stochastic Neighbor Embedding (t-SNE; van der Maaten and Hinton 2008).

3 Cluster visualization

We focus on Neighbor Embedding for cluster visualization. A clustering divides the data objects into a number of groups, such that objects in the same group (called a cluster) are more similar to each other than to those in other groups (clusters) (Tan et al. 2005). The pairwise similarities can be encoded in a similarity matrix p, defined as in NE.

A similarity matrix is considered to be (well) clusterable if there is a clustering such that high similarities appear (much) more probably within clusters than between clusters. If we sort the rows and columns in such a similarity matrix according to the cluster labels, we should observe a diagonal blockwise pattern, where high similarities are denser within the cluster blocks. The pattern is much clearer for a well clusterable similarity matrix.

A good cluster visualization is a display where the user can easily see the groups of data points. In scatter plots, there should be clear space separating the groups such that points in the same group are closer to each other than to those in other groups. Besides sufficient separation, a good cluster visualization should also respect the within-cluster information by approximating the corresponding neighborhoods.

Note that cluster visualization is an unsupervised learning task. No supervised labels are used in finding the coordinates of mapped points, which is different from the supervised/discriminative dimensionality reduction approaches for showing classification boundaries (e.g., Venna et al. 2010; Schulz et al. 2020).

[1] In this paper the matrixwise summation is over off-diagonal elements, i.e. $\sum_{ij} A_{ij} \overset{\text{def}}{=} \sum_{ij:i \neq j} A_{ij}$.

It is often believed that t-SNE can achieve a good cluster visualization (van der Maaten and Hinton 2008) and some successful examples have been presented in the literature. However, as any data analysis method, t-SNE may also produce false-negative results for challenging data sets. That is, t-SNE can result in no apparent clustering for some p matrices, despite them being well clusterable. Such errors are demonstrated with several examples in Sect. 5. The problem cannot be solved by using better optimization algorithms because minimizing the t-SNE objective aims at a better matching of local information and this may not correspond to a better large-scale pattern. A smaller t-SNE cost can in fact correspond to a worse cluster visualization.

4 Neighbor embedding for cluster visualization

In this section, we show in closer detail why t-SNE can fail despite a p matrix being well clusterable and how to correct the problem to obtain good cluster visualizations.

4.1 Generalized stochastic neighbor embedding

We begin with a variant of Kullback–Leibler divergence (Amari 1985) which is defined over non-normalized output similarities with a parameter $s > 0$:

$$D_{\mathrm{I}}(P \mid\mid sq) = \sum_{ij}\left[P_{ij}\ln\frac{P_{ij}}{sq_{ij}} - P_{ij} + sq_{ij}\right]. \tag{1}$$

The divergence is called non-normalized KL-divergence or I-divergence, which measures the discrepancy between P and q at a certain scale s. In this work we focus on the student t-kernel $q_{ij} = (1 + \|y_i - y_j\|^2)^{-1}$.

We call the NE variant that minimizes $D_{\mathrm{I}}(P \mid\mid sq)$ Generalized Stochastic Neighbor Embedding (GSNE) as it generalizes SNE. To see this, we eliminate s by setting $\partial D_{\mathrm{I}}(P \mid\mid sq)/\partial s = 0$, which gives (see Yang et al. 2014 or supplemental document Sect. 1)

$$\arg\min_{Y} D_{\mathrm{KL}}(P \mid\mid Q) = \arg\min_{Y}\min_{s>0} D_{\mathrm{I}}(P \mid\mid sq), \tag{2}$$

where the equality holds when

$$s = \frac{1}{\sum_{ij} q_{ij}}. \tag{3}$$

That is, SNE is a special case of GSNE with s set to the normalizing factor of q.

The SNE choice of s in Eq. 3 is not necessarily an optimal choice for cluster visualization. It aims at matching two neighboring probability matrices. Such a locality preserving objective can prevent the discovery of large-scale patterns, such as clusters present among the input items. Fortunately, below we show that using another choice of s can correct the problem.

4.2 Selecting s for better cluster visualization

The GSNE objective can be rewritten as

$$D_{\mathrm{I}}(P \mid\mid sq) = \mathcal{J}_{\text{attraction}} + \mathcal{J}_{\text{repulsion}} + C \tag{4}$$

where $\mathcal{J}_{\text{attraction}} = -\sum_{ij} P_{ij}\ln q_{ij}$ and $\mathcal{J}_{\text{repulsion}} = s\sum_{ij} q_{ij}$ respectively correspond to attractive and repulsive forces, and $C = -\ln s - 1 + \sum_{ij} P_{ij}\ln P_{ij}$ is a constant w.r.t. q for a given s. Here the scale parameter s also controls the tradeoff between the two forces.

It is known that increasing attraction, e.g., by replacing P with βP ($\beta > 1$) can encourage the mapped data points to form tighter clumps, with more empty space in the visualization (see e.g., van der Maaten and Hinton 2008; van der Maaten 2014; Belkina et al. 2019). The trick is called "early exaggeration" and has been used in t-SNE initialization, where $\beta = 4$ by van der Maaten and Hinton (2008) or $\beta = 12$ by van der Maaten (2014) in the first 250 iterations.[2] The "early exaggeration" trick is equivalent to setting

$$s = \frac{1}{\beta\sum_{ij} q_{ij}} \tag{5}$$

during the initialization. Note that after the initialization, t-SNE still uses $\beta = 1$, i.e. the original choice in Eq. 3. As we will see in Sect. 5 and the supplemental document Sect. 4, the "early-exaggeration" still cannot produce good cluster visualizations, even if $\beta > 1$ is used throughout the iterations (see the supplemental document Sect. 3).

In this work, we propose to choose

$$s = \frac{1}{\sum_{ij} w_{ij} q_{ij}}, \tag{6}$$

where $w_{ij} = \alpha N(N-1)P_{ij} + (1-\alpha)$ with $\alpha \in [0, 1]$ for better cluster visualizations. When $\alpha = 0$, it reduces to the SNE choice. When $\alpha > 0$, it can adaptively bring extra repulsion to improve cluster visualization. To see this, we rewrite the denominator in Eq. 6:

$$\sum_{ij} w_{ij} q_{ij}$$
$$= \sum_{ij}\left[\alpha N(N-1)P_{ij} + (1-\alpha)\right] q_{ij}$$

[2] Other t-SNE optimization methods (e.g. Belkina et al. 2019) may use a different number of iterations spent in early exaggeration.

$$= N(N-1)\left[\alpha \sum_{ij} P_{ij} q_{ij} + (1-\alpha) \sum_{ij} \frac{1}{N(N-1)} q_{ij}\right]$$
$$= N(N-1)\left[\alpha E_1 + (1-\alpha) E_2\right], \tag{7}$$

where the two summation terms in the brackets can be expressed as expectations over different distributions

$$E_1 = \sum_{ij} P_{ij} q_{ij} = \mathbb{E}_{(i,j)\sim\text{Categorical}(P)}\{q_{ij}\} \tag{8}$$

$$E_2 = \sum_{ij} \frac{1}{N(N-1)} q_{ij} = \mathbb{E}_{(i,j)\sim\text{Uniform}([1,\ldots,N]^2)}\{q_{ij}\}. \tag{9}$$

The sampling form enables us to estimate s in a stochastic manner (see details below).

Next, we explain why our choice of s can provide an adaptive attraction-repulsion tradeoff and often lead to a better cluster visualization. When Y is random initially, the two expectations E_1 and E_2 do not differ much, and overall the optimization is similar to the original SNE at the beginning. After minimizing the discrepancy $D_I(P \mid\mid sq)$ for a while, data pairs that correspond to large P_{ij}'s will be mapped closer. As a result, $E_1 > E_2$, which leads to a smaller s and thus smaller repulsion than in SNE. In this work, we simply set $\alpha = 0.5$, and we find it already suffices for many data sets. Because our method often produces clearer clustered displays, we have named it *Stochastic Cluster Embedding* (SCE).

It is important to notice that our work focuses on showing clusters. With $\alpha > 0$, our learning objective does not aim to match the neighborhood probabilities anymore. We intentionally sacrifice some locality-preserving quality (e.g., within-cluster structures) to achieve better cluster visualizations.

4.3 Optimization

The GSNE or SCE objective function is smooth over Y. Therefore it can be optimized by any existing unconstrained smooth optimization techniques such as gradient descent with momentum in t-SNE. However, the original t-SNE algorithm runs in a centralized manner and is thus slow for large-scale data sets. Moreover, the tree-based approximation (Yang et al. 2013; van der Maaten 2014; Vladymyrov and Carreira-Perpiñán 2014) to the objective function and gradient calculation requires rather complex programming (see e.g., Chan et al. 2019).

Here we develop a simple tree-free parallel algorithm to optimize SCE. It repeats the following steps until maximum iterations or the Y change is smaller than a given tolerance:

1. update Y given the current s;
2. estimate s given the current Y,

Because we usually initialize Y with small numbers around zero, all q_{ij}'s are close to 1 at the beginning. Therefore it is reasonable to initialize $s = N(N-1)$.

In both steps, the computation is distributed to a number of computing units called workers. In Step 1, we first rewrite $D_I(P \mid\mid sq)$ in a stochastic form:

$$\mathcal{J}_{\text{attraction}} = \mathbb{E}_{(i,j)\sim\text{Categorical}(P)}\left\{\mathcal{J}^{(i,j)}_{\text{attraction}}\right\}, \tag{10}$$

$$\mathcal{J}_{\text{repulsion}} = \mathbb{E}_{(i,j)\sim\text{Uniform}([1,\ldots,N]^2)}\left\{\mathcal{J}^{(i,j)}_{\text{repulsion}}\right\}, \tag{11}$$

where $\mathcal{J}^{i,j}_{\text{attraction}} = -\ln q_{ij}$ and $\mathcal{J}^{(i,j)}_{\text{repulsion}} = sN(N-1)q_{ij}$. According to this form, each worker randomly draws a pair (i, j) for attraction and another pair for repulsion, calculates their partial stochastic gradients w.r.t. y_i and y_j:

$$\frac{\partial \mathcal{J}^{(i,j)}_{\text{attraction}}}{\partial y_i} = -\frac{\partial \mathcal{J}^{(i,j)}_{\text{attraction}}}{\partial y_j}$$
$$= -2q_{ij}(y_i - y_j), \tag{12}$$

$$\frac{\partial \mathcal{J}^{(i,j)}_{\text{repulsion}}}{\partial y_i} = -\frac{\partial \mathcal{J}^{(i,j)}_{\text{repulsion}}}{\partial y_j}$$
$$= 2sN(N-1)q^2_{ij}(y_i - y_j), \tag{13}$$

and updates the corresponding mapped points by stochastic partial gradient descent. In this way, each worker requires only $O(1)$ cost for updating a pair of mapped points y_i and y_j.

Next we consider how to estimate s in an asynchronously stochastic and distributed manner. The (i, j) samples and the corresponding q_{ij}'s in the denominator of Eq. 6 (i.e., s^{-1}), have already been obtained from Step 1. Denote by ξ and ω, respectively, the weighted sum and count of newly calculated q_{ij}'s. We get $\frac{N(N-1)\xi}{\omega}$ as a stochastic approximation of s^{-1} and mix it with the current estimate as the new one:

$$s^{-1} \leftarrow \rho s^{-1} + (1-\rho)\frac{N(N-1)\xi}{\omega}, \tag{14}$$

where $\rho \in (0, 1)$ is the forgetting rate. We find $\rho = \frac{N(N-1)}{N(N-1)+\omega}$ working well in practice.

The pseudo-code of the SCE algorithm is given in Algorithm 1. Because the algorithm is almost lock-free, it is straightforward to implement it efficiently on multi-core CPUs and GPUs. Our algorithm belongs to the family of stochastic block coordinate descent optimization, for which Richtárik and Takáč (2011) gave the convergence guarantee and convergence rate.

Algorithm 1 Stochastic Cluster Embedding

```
Input: similarity matrix $P$, number of iterations $T$.
Initialize $Y = \{y_i\}_{i=1}^N$ with small numbers, e.g. $y_{id} \sim \mathcal{N}(0, 10^{-4})$
    for all $i, d$.
Initialize $s^{-1} \leftarrow N(N-1)$
for t=0 to T do
    $\eta_t \leftarrow 1 - t/T$
    $\xi \leftarrow 0$
    $\omega \leftarrow 0$
    parallel for each worker do
        Draw $(i, j) \sim$Categorical($P$)                ▷ handle attraction
        $q_{ij} \leftarrow (1 + \|y_i - y_j\|^2)^{-1}$
        $\nabla \leftarrow -2q_{ij}(y_i - y_j)$
        $y_i \leftarrow y_i + \eta_t \nabla$    $y_j \leftarrow y_j - \eta_t \nabla$
        $\xi \leftarrow \xi + \alpha q_{ij}$    $\omega \leftarrow \omega + \alpha$
        Draw $(i, j) \sim$Uniform($[1, \ldots, N]^2$)     ▷ handle repulsion
        $q_{ij} \leftarrow (1 + \|y_i - y_j\|^2)^{-1}$
        $\nabla \leftarrow 2sN(N-1)q_{ij}^2(y_i - y_j)$
        $y_i \leftarrow y_i + \eta_t \nabla$    $y_j \leftarrow y_j - \eta_t \nabla$
        $\xi \leftarrow \xi + (1-\alpha)q_{ij}$    $\omega \leftarrow \omega + (1-\alpha)$
    end for
    $\rho = \frac{N(N-1)}{N(N-1)+\omega}$
    $s^{-1} \leftarrow \rho s^{-1} + (1-\rho)\frac{N(N-1)\xi}{\omega}$;
end for
Output: low-dimensional representations $Y$
```

5 Experiments

We have compared our method with t-SNE, as well as two other more recent methods called LargeVis and UMAP:

- t-SNE: We have used the implementation[3] by van der Maaten (2014), where the maximum iterations in t-SNE was set to 10,000 (ten times as the default) to get closer to convergence.
- LargeVis (Tang et al. 2016): we have used its official implementation in GitHub.[4]
- UMAP (Uniform Manifold Approximation and Projection; McInnes et al. 2018): we have used the `umap-learn` package in Python.

We have used six real-world data sets in our experiments:

- `IJCNN`: the IJCNN 2001 neural network competition data.[5] There are 126,701 samples of 22 dimensions and from ten engines (classes).
- `TOMORADAR`: The data was collected via a helicopter-borne microwave profiling radar (Chen et al. 2017) termed FGI-Tomoradar to investigate the vertical topography structure of forests. After preprocessing, the data set contains 120,024 samples of 8192 dimensions from three classes.
- `FLOW-CYTOMETRY`: the single-cell biology data set collected from Flow Repository.[6] After preprocessing, the data set contains 1,000,000 samples of 17 dimensions.
- `HIGGS`: the HIGGS Data Set in the UCI repository.[7] The data was produced using Monte Carlo simulations of the particles in a physics experiment. There are 11,000,000 data points of 28 dimensions. Previously the data were used for classification between the bosons and the background particles, whereas there is little research on unsupervised learning on the data. Here we compared visualizations to discover the particle clusters.
- `SHUTTLE`: the Statlog (Shuttle) Data Set in the UCI repository.[8] There are 58,000 samples of 9 dimensions in three large and four small classes.
- `MNIST`: the MNIST database of handwritten digits.[9] There are 70,000 samples of 784 dimensions in ten digit classes.

To our knowledge, it is a largely unsolved problem how to convert vectorial data to a similarity matrix optimally, and there is no universally best solution. In practice, popular choices are Entropic Affinity (EA; Hinton and Roweis 2003) and k-Nearest Neighbor (k-NN) with tuning of the perplexity (or k) parameter.

This work focuses on cluster embedding of a given similarity matrix P.[10] We have constructed the P matrix by using EA with perplexity 30 for `SHUTTLE` and `IJCNN`. We have used symmetrized k-NN graph adjacency matrix as P for `MNIST`, `TOMORADAR`, `FLOW-CYTOMETRY` and `HIGGS` with $k = 10$, $k = 50$, $k = 15$ and $k = 5$, respectively. In this way, the constructed similarity matrices are well clusterable because they comprise a diagonal blockwise pattern, as we shall see in Fig. 8. A good cluster visualization method should be able to show these clusters clearly.

We have performed three groups of empirical studies to verify the advantages of the proposed SCE method. Below we first demonstrate its visualizations compared with those by t-SNE, LargeVis, and UMAP. Second, we compare the s choices in SCE and t-SNE to our user study. Third, we verify the clustering quality of the compared methods by seeing how well they can group the input similarities.

[3] https://github.com/lvdmaaten/bhtsne.

[4] https://github.com/lferry007/LargeVis.

[5] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html.

[6] https://flowrepository.org/id/FR-FCM-ZZ36.

[7] https://archive.ics.uci.edu/ml/datasets/HIGGS.

[8] https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle).

[9] http://yann.lecun.com/exdb/mnist/.

[10] Our method is unsupervised. We did not use supervised labels in constructing the similarity matrices.

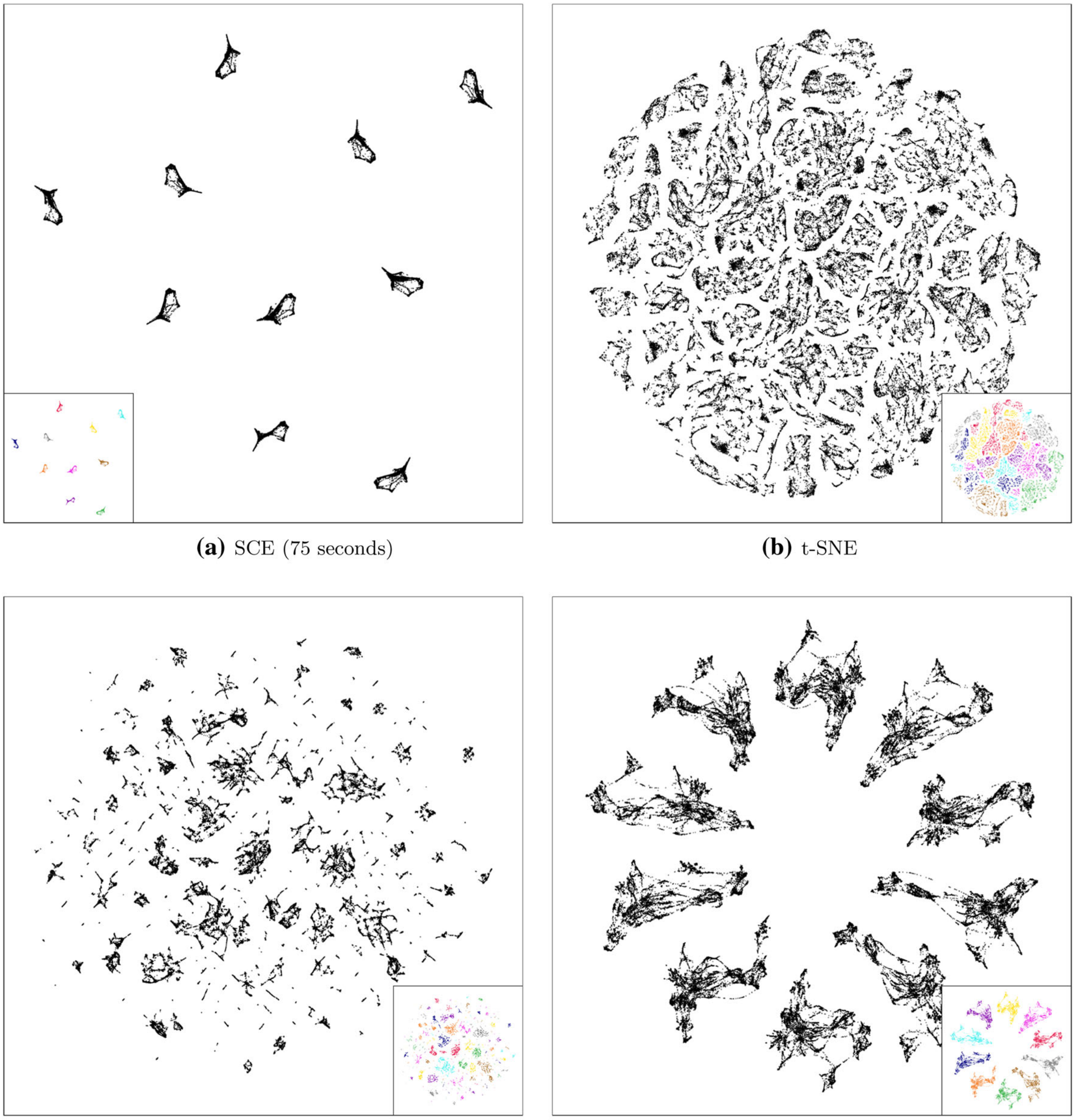
5.1 Visualization comparison

The SCE visualizations compared with the other methods are shown in Figs. 1, 2, 3, 4, 5 and 6. It is important to notice that cluster visualizations are unsupervised. Therefore, we should focus on the colorless scatter plots and check whether large-scale patterns (i.e. clusters) appear or not. The colors in the sub-figures are only used for reference to verify the alignment between clusters and ground truth classes.

We can see that SCE works well for all six data sets. Compared with other approaches, SCE shows several typically compact clusters with clear separation among them. Therefore it is easy to see the major clusters in all even if color is removed (i.e. without supervised labels).

t-SNE fails for five of the six test data sets. For SHUTTLE, IJCNN, TOMORADAR, the t-SNE layouts overall look like a single diamond with too many small groups, where no major clusters can be identified. For FLOW-CYTOMETRY the t-SNE visualization is nearly a single ball, while for HIGGS it is just a hairball. The only barely successful case is MNIST, but there are still many data points that scatter between the clusters, leaving the boundaries unclear.

LargeVis is slightly better than t-SNE. It correctly shows ten clusters for MNIST. In the LargeVis visualizations of SHUTTLE, IJCNN, TOMORADAR and FLOW-CYTOMETRY,

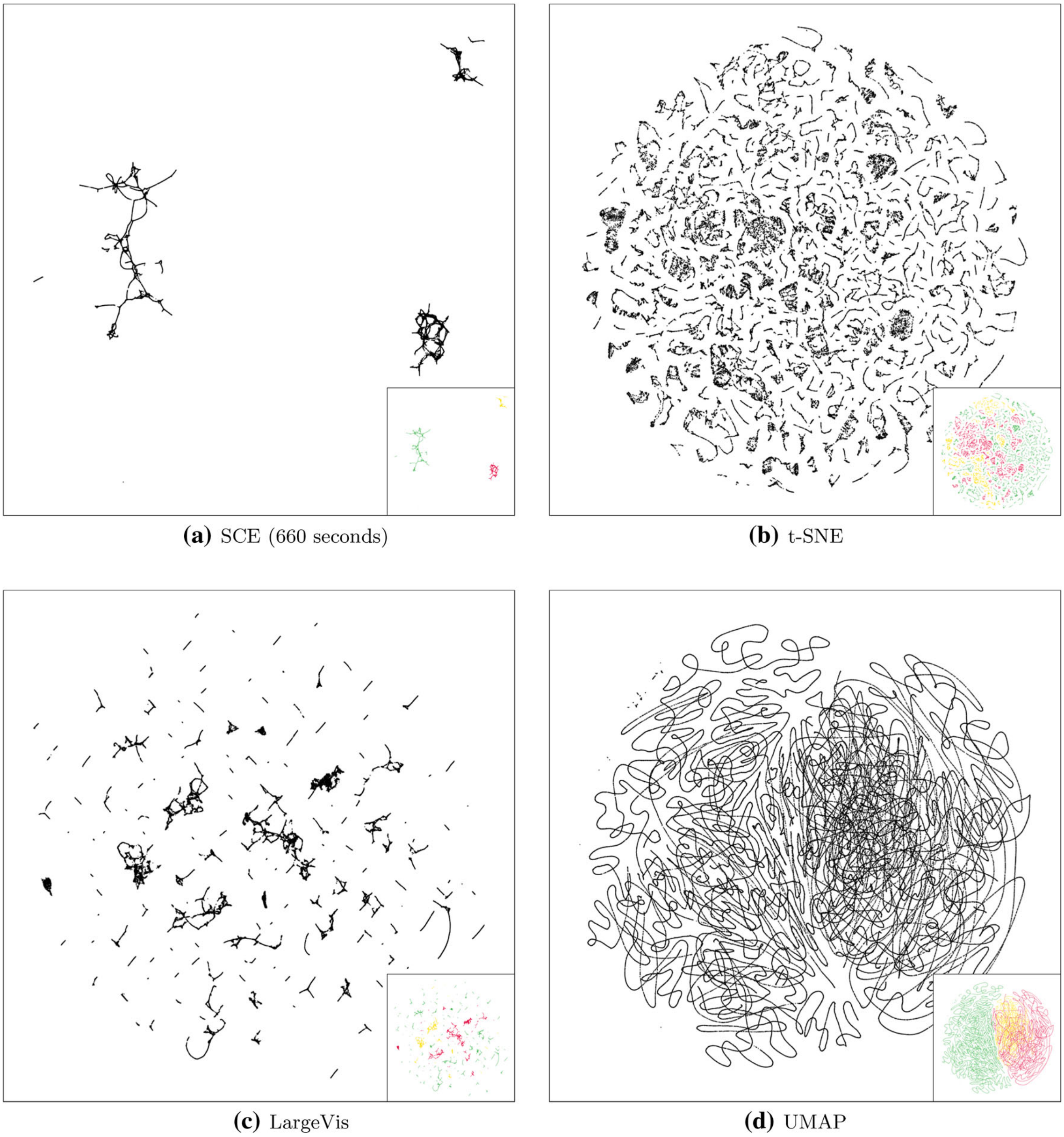

(a) SCE (75 seconds)

(b) t-SNE

(c) LargeVis

(d) UMAP

Fig. 1 Visualizations of the IJCNN data set by using the compared methods. The classes are shown by colors in the small sub-figures



(a) SCE (660 seconds)

(b) t-SNE

(c) LargeVis

(d) UMAP

Fig. 2 Visualizations of the TOMORADAR data set by using the compared methods. The classes are shown by colors in the small sub-figures

we can also see some groups of data points. However, there are still too many small groups, and it is hard to identify the major clusters. LargeVis fails for HIGGS, where no clear cluster is shown.

UMAP works better than t-SNE and LargeVis for some data sets. It also correctly shows ten clusters for MNIST. UMAP can separate the ten engine clusters for IJCNN. We can barely see several major clusters in its FLOW-CYTOMETRY visualization. The method also successfully identifies several clusters for HIGGS. However, UMAP does not work well for SHUTTLE because there are many small groups without a clear separation of major clusters. UMAP fails for TOMORADAR, where no clustering pattern is found.

5.2 User study

Since visualizations are designed for human use, we have performed a user study about the human choice among GSNE visualizations corresponding to a range of s values for seeing clusters. We can then compare the resulting s values in SCE and t-SNE to see which is closer to the human choices.

We have used the four smallest data sets IJCNN, TOMORADAR, SHUTTLE, and MNIST. For each data set, we have ran GSNE with $s = 10^t \cdot N^{-2}$, where $t \in [-4, 6]$ for TOMORADAR and $t \in [0, 8]$ for the other data sets. These ranges of s values were set to be wide enough from over-attractive to over-repulsive such that meaningful s choices should take place in between.

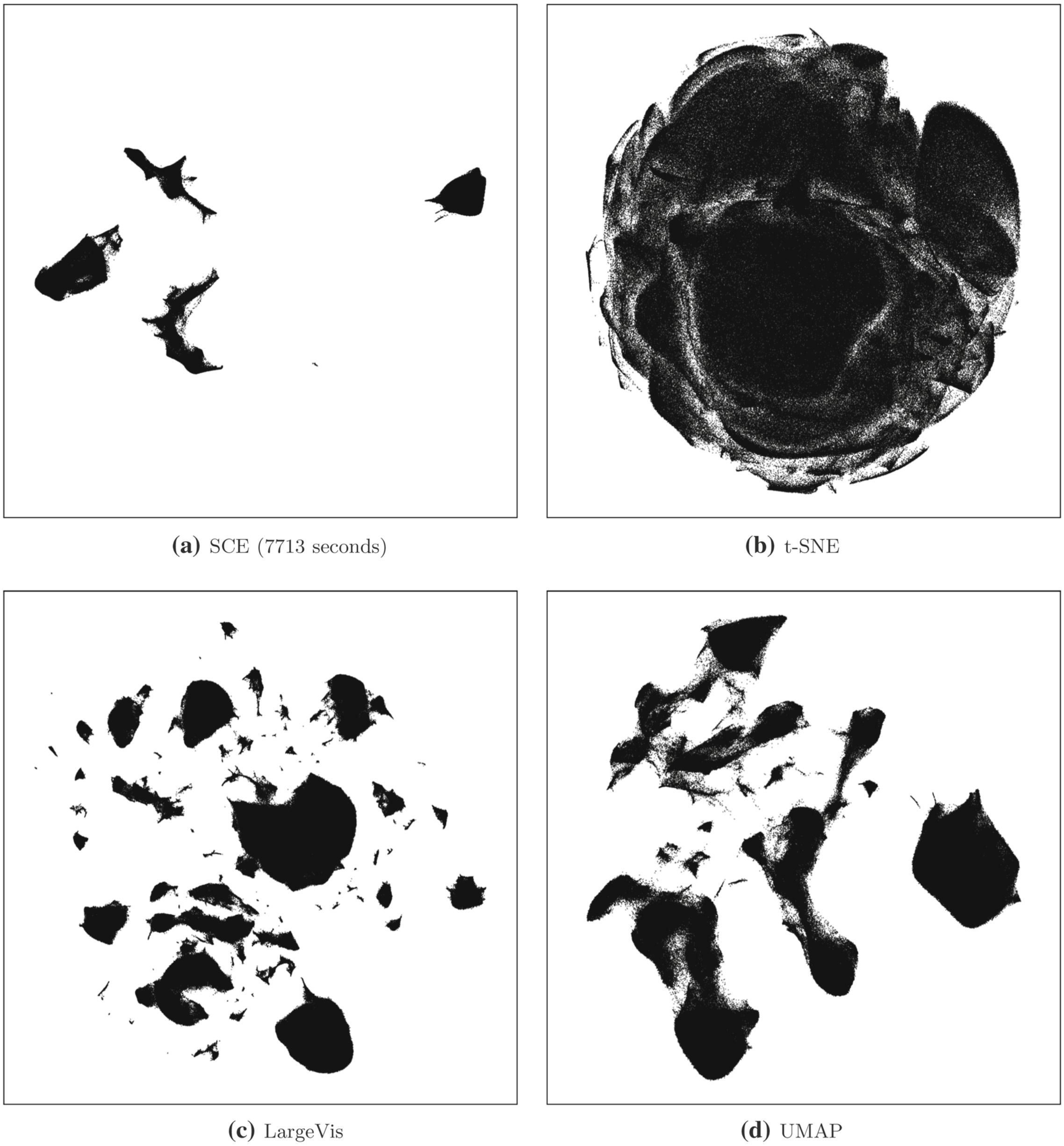


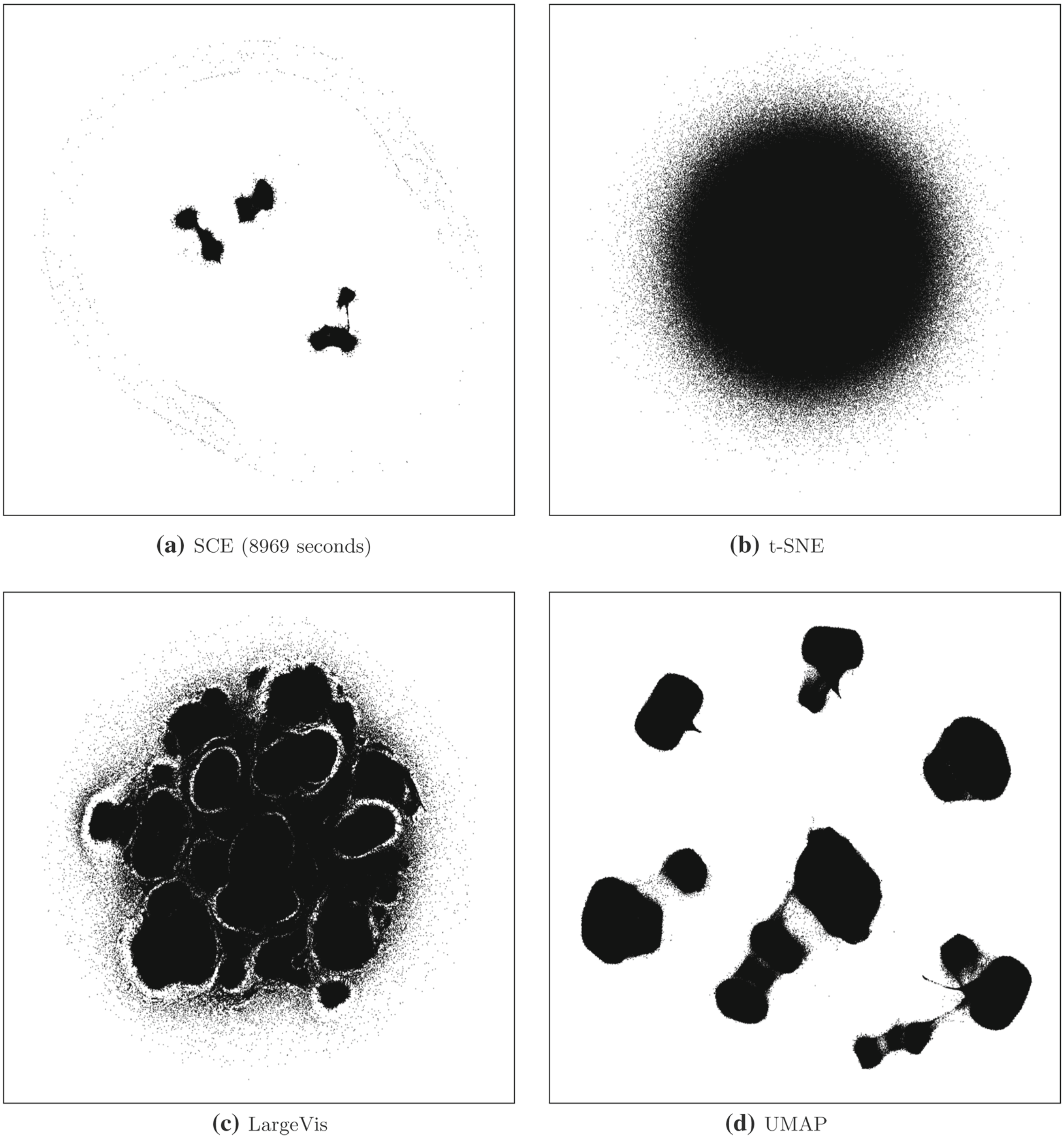
Fig. 3 Visualizations of the `FLOW-CYTOMETRY` data set by using the compared methods

The user interface of the study can be found in http://clres.cs.hut.fi/ClAnalysis/webpage.html. For each data set, the series of visualizations are presented to a user (see the supplemental document Sect. 2 for a screenshot), where he or she uses a slider to specify the s value and inspects the corresponding pre-computed visualization. The user selects a preferred s value for cluster visualization and then presses the "Next" button. The system records the user choice, and the study proceeds to the next series of visualization.

We first performed a controlled laboratory study, where 40 users came to the test computer room and gave their evaluations. We later conducted a crowdsourcing user study, following the established good practices of crowdsourcing for visualization research (Borgo et al. 2017). Using the crowdsourcing platform CrowdFlower,[11] we collected empirical data from a large and diverse population made up of 300 participants. We then combined the data of the controlled and crowdsourcing studies for the analysis, leading to 340 answers for each data set. More details about the user study can be found in the supplemental document, Sect. 2.

The results are shown in Fig. 7. The SNE choice of s according to Eq. 3 is shown by blue dotted-dash lines. We can see that the s chosen by SNE is on the right of the human median (solid green line) for all data sets, which indicates

[11] https://www.crowdflower.com/, now part of Figure-eight.


(a) SCE (8969 seconds)

(b) t-SNE

(c) LargeVis

(d) UMAP

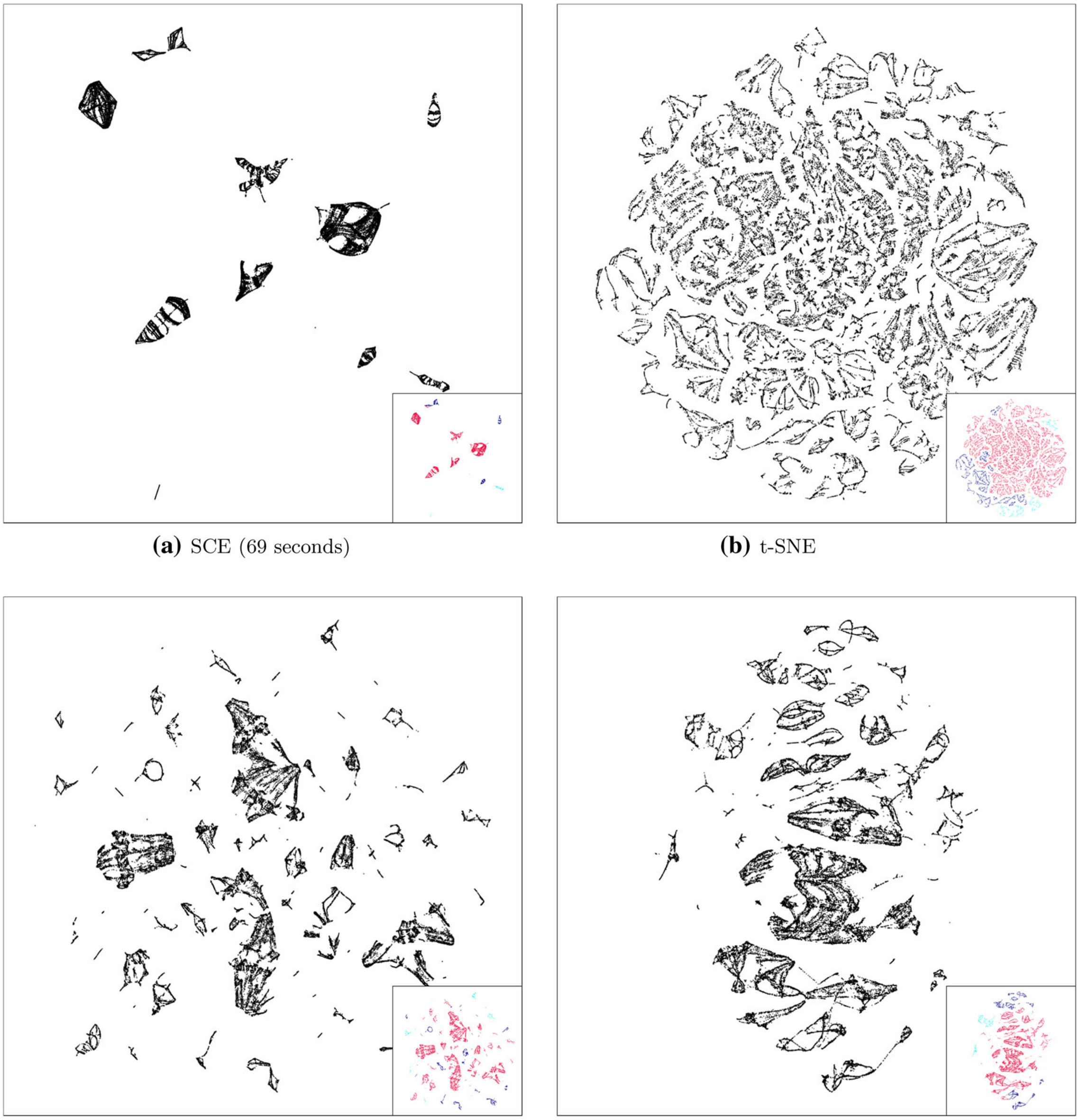
Fig. 4 Visualizations of the HIGGS data set by using the compared methods

that, in human eyes, GSNE with a smaller s is often better than t-SNE for cluster visualization. In contrast, the SCE choice (red dash lines) are closer to the human median for all four data sets.

5.3 Clustering quality

Cluster visualization is an unsupervised task where the supervised labels are not available. Even if class labels are available in some data sets, they are not necessarily aligned with the intrinsic data clusters, for example, in SHUTTLE and HIGGS.

Here we have used an unsupervised approach to verify the SCE clustering quality. We first manually clustered the mapped data points in 2D space. Because most clusters are well separated in the SCE visualizations, the manual clustering is easy with little ambiguity. After the clustering, we reordered the rows and columns of the input P matrix according to the cluster labels and examined the nonzero entries (blue dots) in Fig. 8. We can see blockwise diagonal patterns in all plots, where each block corresponds to a cluster in the visualizations. Moreover, the blue dots within clusters are denser than those between clusters, which means SCE achieves good clustering quality.



(a) SCE (69 seconds)

(b) t-SNE

(c) LargeVis

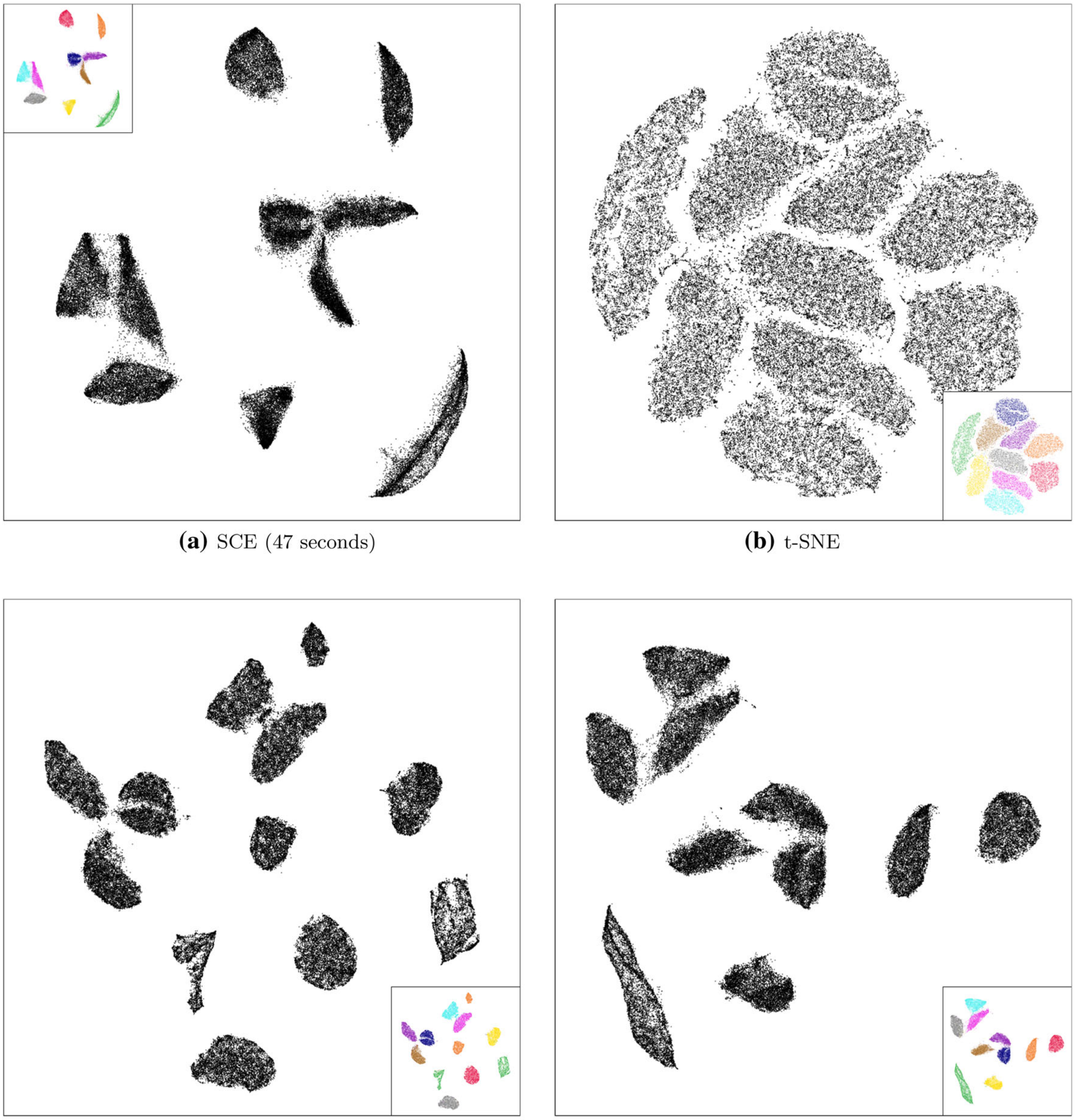
(d) UMAP

Fig. 5 Visualizations of the SHUTTLE data set by using the compared methods. The classes are shown by colors in the small sub-figures. (Color figure online)

6 Conclusions

We have presented a new nonlinear dimensionality reduction method called Stochastic Clustering Embedding for better cluster visualization. Our method modifies t-SNE by using an adaptive and effective attraction-repulsion tradeoff. We have tested our method in various real-world data sets and compared it with other modern NLDR methods. The experimental results show that our method can consistently identify the intrinsic clusters. Furthermore, we have contributed a simple and fast optimization algorithm that can easily be implemented in modern parallel computing platforms.

In this work, we have only considered the layout algorithms which produce the embedding coordinates. The visualization quality is also determined by the visual elements such as dot sizes, colors, and opacity in the display. One promising area for further research would be to incorporate a cognitive user model, which could potentially improve cluster visualization to a significant degree. Such models could be fitted with Approximate Bayesian Computation as shown by e.g., Kangasrääsiö et al. (2017); Micallef et al. (2017) using the efficient implementation of inference algorithms available in the ELFI Python package (Lintusaari et al. 2018).



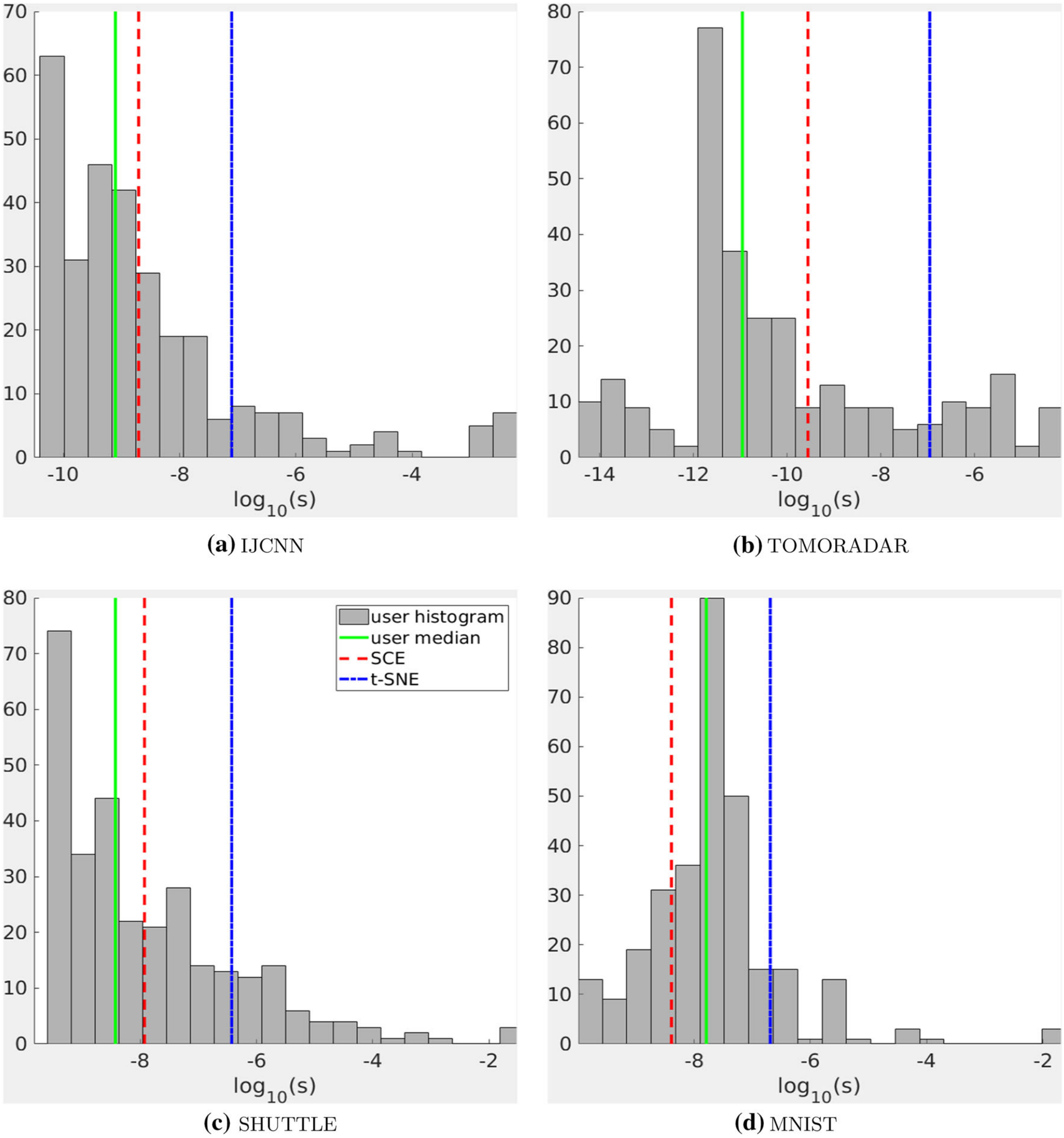
(a) SCE (47 seconds)

(b) t-SNE

(c) LargeVis

(d) UMAP

Fig. 6 Visualizations of the MNIST data set by using the compared methods. The classes are shown by colors in the small sub-figures. (Color figure online)



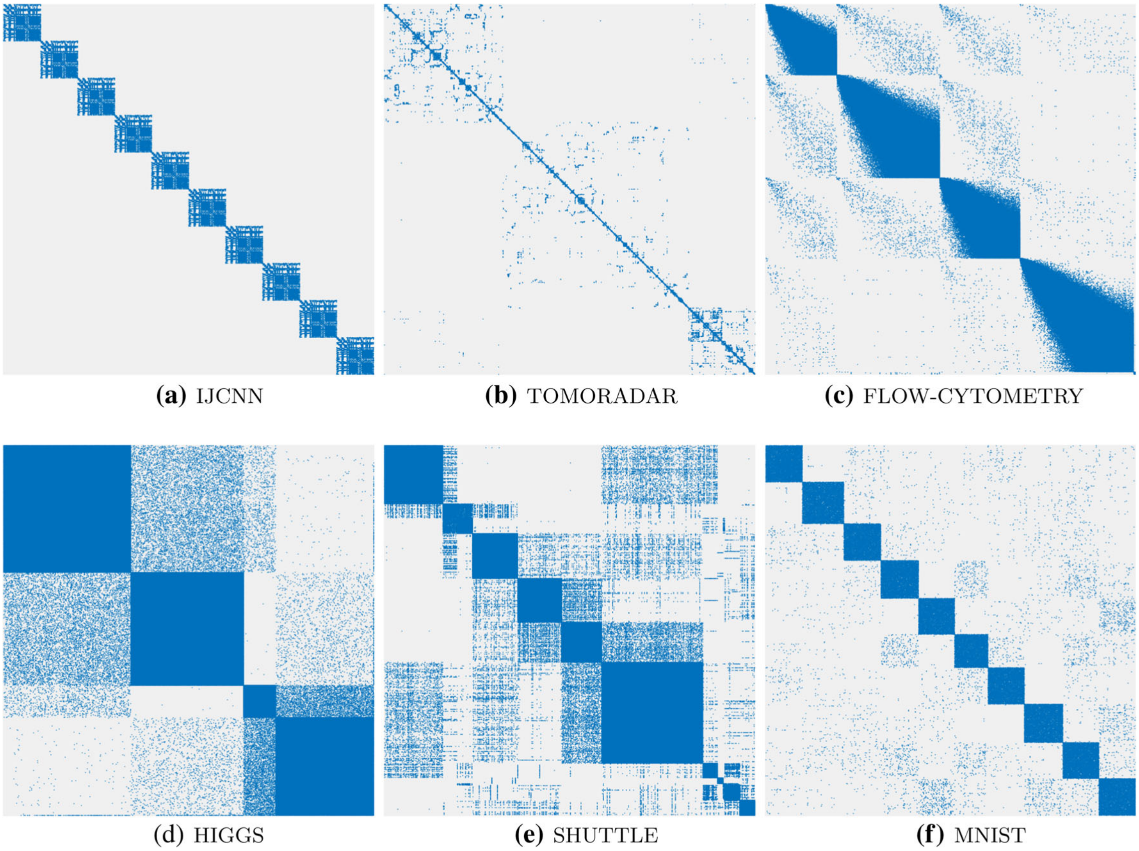
(a) IJCNN

(b) TOMORADAR

(c) SHUTTLE

(d) MNIST

Fig. 7 The s values for cluster visualization: (gray bars) histogram of human choices, (green solid line) median of human choices, (red dash line) SCE choice, and (blue dash dotted line) t-SNE choice. (Color figure online)


(a) IJCNN **(b)** TOMORADAR **(c)** FLOW-CYTOMETRY

(d) HIGGS **(e)** SHUTTLE **(f)** MNIST

Fig. 8 Visualization of the similarity matrix P of the experimented data sets using Matlab `spy` function, where the rows and columns are sorted by the manual cluster labels. Blue dots show the 1's in the matrix and white dots show the 0's. Due to limited resolution, the figures shows a uniform subsample 10% data points. (Color figure online)

Acknowledgements This work was supported by The Research Council of Norway, Grant Number 287284, ERC Grant Number 742158, the Academy of Finland (Flagship programme: Finnish Center for Artificial Intelligence FCAI), and UKRI Turing AI World-Leading Researcher Fellowship, EP/W002973/1. We acknowledge for using the IDUN computing cluster (Själander et al. 2019) provided at Norwegian University of Science and Technology.

Funding Open access funding provided by NTNU Norwegian University of Science and Technology (incl St. Olavs Hospital - Trondheim University Hospital).

Declarations

Conflict of interest Potential conflicts of interests in the reviewing process: ntnu.no, aalto.fi, helsinki.fi, uio.no, nsl.fi, and sanger.ac.uk. The research involved a user study on the s-value choice in GSNE. All participants were informed about the tasks. We collected only results from those who consented to the tasks. No identifying information or personal privacy was recorded in the user study.

References

Amari, S.: Differential-Geometrical Methods in Statistics. Springer, Berlin (1985)

Belkina, A., Ciccolella, C., Anno, R., Halpert, R., Spidlen, J., Snyder-Cappione, J.: Automated optimized parameters for t-distributed stochastic neighbor embedding improve visualization and analysis of large datasets. Nat. Commun. **10**(5415), 1–12 (2019)

Borgo, R., Lee, B., Bach, B., Fabrikant, S., Jianu, R., Kerren, A., Kobourov, S., McGee, F., Micallef, L., von Landesberger, T., Ballweg, K., Diehl, S., Simonetto, P., Zhou, M.: Crowdsourcing for information visualization: Promises and pitfalls. In: Archambault, D., Purchase, H., Hoßfeld, T. (Eds.) Evaluation in the Crowd. Crowdsourcing and Human-Centered Experiments, Cham, Springer International Publishing. pp. 96–138 (2017). ISBN 978-3-319-66435-4

Chan, D.M., Rao, R., Huang, F., Canny, J.F.: Gpu accelerated t-distributed stochastic neighbor embedding. J. Parallel Distrib. Comput. **131**, 1–13 (2019)

Chen, Y., Hakala, T., Karjalainen, M., Feng, Z., Tang, J., Litkey, P., Kukko, A., Jaakkola, A., Hyyppä, J.: Uav-borne profiling radar for forest research. Remote Sens. **9**(1), 58 (2017)

Hinton, G., Roweis, S.: Stochastic neighbor embedding. In: Advances in Neural Information Processing Systems (NIPS), pp. 857–864 (2003)

Kangasrääsiö, A., Athukorala, K., Howes, A., Corander, J., Kaski, S., Oulasvirta, A.: Inferring cognitive models from data using approximate Bayesian computation. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI), pp. 1295–1306 (2017)

Lintusaari, J., Vuollekoski, H., Kangasrääsiö, A., Skytén, K., Järvenpää, M., Marttinen, P., Gutmann, M.U., Vehtari, A., Corander, J., Kaski, S.: Elfi: engine for likelihood-free inference. J. Mach. Learn. Res. **19**(16), 1–7 (2018)

McInnes, L., Healy, J., Melville, J.: UMAP: uniform manifold approximation and projection for dimension reduction. arXiv e-prints (2018)

Micallef, L., Palmas, G., Oulasvirta, A., Weinkauf, T.: Towards perceptual optimization of the visual design of scatterplots. IEEE Trans. Vis. Comput. Gr. **23**(6), 1588–1599 (2017)

Richtárik, P., Takáč, M.: Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. Math. Program. **144**(1–2), 1–38 (2011)

Schulz, A., Hinder, F., Hammer, B.: Deepview: visualizing classification boundaries of deep neural networks as scatter plots using discriminative dimensionality reduction. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI), pp. 2305–2311 (2020)

Själander, M., Jahre, M., Tufte, G., Reissmann, N.: EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure (2019)

Tan, P., Steinbach, M., Karpatne, A., Kumar, V.: Introduction to data mining. Addison Wesley, Boston (2005)

Tang, J., Liu, J., Zhang, M., Mei, Q.: Visualizing large-scale and high-dimensional data. In: Proceedings of International Conference on World Wide Web (WWW), pp. 287–297 (2016)

van der Maaten, L.: Accelerating t-SNE using tree-based algorithms. J. Mach. Learn. Res. **15**, 3221–3245 (2014)

van der Maaten, L., Hinton, G.: Visualizing high-dimensional data using t-SNE. J. Mach. Learn. Res. **9**, 2579–2605 (2008)

Venna, J., Peltonen, J., Nybo, K., Aidos, H., Kaski, S.: Information retrieval perspective to nonlinear dimensionality reduction for data visualization. J. Mach. Learn. Res. **11**, 451–490 (2010)

Vladymyrov, M., Carreira-Perpiñán, M.: Linear-time training of nonlinear low-dimensional embeddings. In: Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS), pp. 968–977 (2014)

Yang, Z., Peltonen, J., Kaski, S.: Scalable optimization of neighbor embedding for visualization. In: Proceedings of International Conference on Machine Learning (ICML), pp. 127–135 (2013)

Yang, Z., Peltonen, J., Kaski, S.: Optimization equivalence of divergences improves neighbor embedding. In: Proceedings of International Conference on Machine Learning (ICML), pp. 460–468 (2014)