Constraint-Preserving Scores for Automatic Hyperparameter Tuning of Dimensionality Reduction Methods for Visualization

Viet Minh Vu, Adrien Bibal and Benoît Frénay

NADI Institute - PReCISE Research Center, Faculty of Computer Science, University of Namur, Rue Grandgagnage 21, 5000 Namur, Belgium

ARTICLE INFO

Keywords:
Machine Learning
Dimensionality Reduction
Visualization
Bayesian Optimization
Pairwise Constraints

ABSTRACT

1. Introduction

(1) Keep the section 1 (context, problematic).

(2) Motivation: Why we choose this approach of automatically tuning the hyperparameter without modifying the choosen DR methods?

+ Easy to adapt to the existing DR methods.

+ Towards AutoML (need citation) but can keep the explainability. Our method not only find the optimal visualization but also explain why it is.

(3) Add a small paragraph to introduce several visualization methods which are widely used in practice but hard to tune the (hyper)-hyperparameters: tSNE [18], LargeVis [27], UMAP [19].

(4) Our solution:

+ Constraint preserving score to measure the similarity preserving in the visualization. (TODO need to find the goal of our score, and say why it is worth to measure the similarity in the viz).

+ Bayesian Optimization (BayOpt) approach [21, 4] for hyperparameter tuning.

(5) Main contributions: TODO: Complete later.

To Discuss: What is exactly our goal: tune the hyperparameter (w.r.t. a score/metric) or to find the best visualization (w.r.t. to the evaluation of the real user for example)?

(6) The target audiences:

+ The end-users who want to apply the visualization methods to their own data without caring about the complex algorithms and hyperparameters. They can use our method as a blackbox hyperparameter tuning toolbox with an additional price of providing the labels or a partial of the labels for the dataset. (Refer to the section analyzing the impact of the number of constraints).

+ The experts who want to analze the impact of the hyperparameters and to evaluate the quality of the visualization. They can use our method as a transparent toolbox to understand the internal step in the optimization process thanks to BayOpt approach.

2. Background and Related Work

2.1. Visualization Quality Metrics

There exist several metrics to evaluate the quality of an embedding. This section presents five of these metrics that we use for evaluating the embedding selected by our proposed method. In this paper, among the possible quality measures, we do not consider clustering quality measures because cluster metrics need labeled data for measurement (e.g. the image-related number in the dataset MNIST), which would restrict the datasets on which we can evaluate the user constraints approach. For these reasons, we only use cluster-label agnostic metrics to measure the quality of the *t*-SNE embeddings. We also avoid quality measures that are linked to the objective function of SNE, e.g. Neighborhood Retrieval Visualizer (NeRV) [29], (i) because they need the evaluation of their (own) perplexity parameter and (ii) because of the bias of measuring the quality of *t*-SNE with a quality metric too closely related to SNE.

Five metrics have been selected for evaluating the quality of visualizations. The first considered metric, the *correlation coefficient* (CC) [12], compares the pairwise distances in the HD and LD spaces by computing the correlation between the distance vectors in HD and LD. The well-known *Kruskal's non-metric stress* (NMS) [14], often used as objective function of non-metric multidimensional scaling, is used to compare the pairwise distance orders between the high and low-dimensional space. The *curvilinear component analysis stress* (CCA) [10] is a kind of Kruskal's stress with an emphasis on the embedding pairwise distances. The metric evaluates the embedding quality by looking if instances in the low-dimensional space are close to each other. The *Sammon's non-linear mapping stress* (NLM) [23], is a mea-

*Corresponding author
ORCID(s):

Table 1
Properties of the five cluster-label-agnostic quality metrics considered in this paper to assess visualizations.

Metric name	Range value	Description
CC	$[0, 1]$	Pearson correlation coefficient between pairwise distance vectors
NMS	$[0, +\infty)$	Stress based on comparison of pairwise distance orders
CCA	$[0, +\infty)$	Stress with accent put on low dim.
NLM	$[0, +\infty)$	Stress with accent put on high dim.
AUC_{log}RNX	$[0, 1]$	How neighbors in high dim. are preserved in low dim.

sure similar to CCA, but focusing on the closeness of instances in the high-dimensional space. Finally, the $AUC_{log}RNX$ [16] compares the neighborhood of each instance in the high and low-dimensional spaces for all possible neighborhood sizes. Table 1 summaries these metrics and mathematical details are provided in Appendix A.

2.2. Usage of Pairwise Constraints in Unsupervised Learning

2.2.1. User Constraints for Clustering

Clustering is a machine learning problem in which the goal is to find groups (called *clusters*) of instances in the data. The constraints used in clustering methods have been well studied for a long time. User constraints can incorporate domain expertise with the goal of explicitly defining the property of the expected clusters. The popular survey by Davidson et al. [8] focuses on *constraint-based* and *distance-based* clustering methods with instance-level constraints. In constraint-based methods, the clusters are formed in such a way that the given constraints are preserved as much as possible, e.g. PCKMeans [2] or a modified version of K-Means with feasibility under $\delta-$, $\epsilon-$ and pairwise constraints [9]. In distance-based methods, the constraints are first used to train a distance function that is later used by a clustering algorithm, e.g. relation component analysis as distance measure [1] and constraint metric K-Means [32].

The pairwise constraints are first introduced in constrained K-Means by Wagstaff et al. [30] for clustering GPS data. Must-link and cannot-link constraints indicate that two instances must be in the same cluster or cannot be in the same cluster, respectively. In this paper, these terms are generalized for more widely usages. A similar-link constraint suggests that two points should be close or similar and a dissimilar-link constraint means that two points should be far apart or dissimilar.

2.2.2. User Constraints for Dimensionality Reduction

One application of user constraints in DR methods is for visualizing data in which we can inject constraints to force the output embedding to have some expected properties. The *objective constraints* can be partial labels (as in semi-supervised LDA [26]) or constraints on the value of features (e.g. bounded PCA [13]). If users can interact with the visualization result, they can give their feedbacks in form of instance-level *subjective constraints*. Integrating user interaction into DR methods is reviewed by Sacha et al. [22] and by Endert et al. in a wider survey on integrating machine learning into visual analysis [11]. Pairwise constraints from user feedbacks are used to attract points connected by similar-links and repulse dissimilar-link-constrained points. Such constraints are used in e.g. pairwise constraints-guided feature projection [28], semi-supervised DR [33], graph-driven constrained DR via linear projection [7] and constrained locality preserving projections [6].

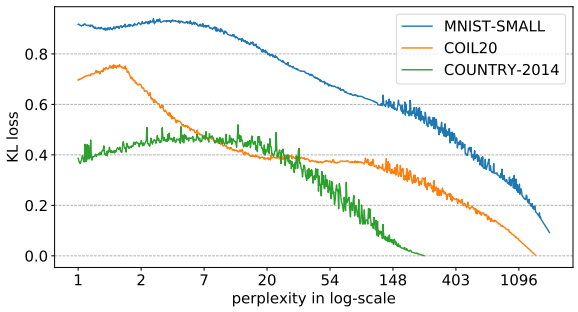


Figure 1: For three different datasets, the KL loss tends to decrease when the perplexity increases. Note that the values of KL loss across different datasets are not comparable.

2.3. Choosing Hyperparameters for DR methods

TODO: Review

The right perplexity to choose depends on the characteristics of the dataset such as the number of instances (size), the topology (structure) or the density (distribution) of instances, which makes it hard to select the best one. The original paper of van der Maaten et al. [18] suggests typical values between 5 and 50. However, in practice, the embedding can change drastically between two different perplexity values. Therefore, there is no evidence to ensure that the suggested perplexities are good for all datasets. The original paper also proposes a simple method to select a good perplexity by looking at the KL loss produced by several perplexities and choose the lowest one, since it corresponds to a well-preserved neighborhood. However, the KL loss tends to naturally decrease when the perplexity increases [5], which is confirmed by our experiments as shown in Fig. 1. For this reason, it is unsuitable to use the KL loss for evaluating the embedding quality since a very high perplexity would be chosen.

In practice, users have to carefully choose a hard-to-understand perplexity to obtain a good embedding result. Even though this process requires an expertise in machine learning, it is often tedious and error-prone. Our idea is to let users suggest *how close instances have to be in the visualization* by using constraints, in order to propose the most suitable visualization for them.

Few papers in the literature attempt to derive the best perplexity automatically. Strickert [25] evaluates the degree of neighborhood P*, instead of P, by using a given pairwise score matrix S. For each instance i in $\mathbf{S}$, all other instances j are ranked and P^*_{ij} is computed with respect to the ranking position of each instance j in $\mathbf{S}_i$. However, this solution is not suitable when the matrix $\mathbf{S}$ is not provided. Lee et al. [15] use a multi-scale approach by considering, and averaging, all neighborhood sizes. Despite providing visualizations by bypassing the perplexity selection problem, Lee et al. do not offer a solution for the selection problem itself. On the contrary, Cao and Wang [5] select the perplexity that minimizes the modified *Bayesian Information Criteria* (*BIC*):

$$S(perplexity) = 2KL(P||Q) + \log(n)\frac{perplexity}{n}, \quad (1)$$

where $KL(P||Q)$ is the same KL loss as Eq. **??** and n is the number of instances. Their approach is validated with user-based experiments by comparing the automatically selected visualization to the ones selected by users. They obtain resultant perplexities very close to the user consensus for three datasets.

2.4. Automatic Hyperparameter Tuning with Bayesian Optimization

The machine learning methods in general are controlled by one or many hyperparameters. To use these methods efficiently, it requires a lot of skill to set hyperparameters. The efficiency of a method is usually evaluated by a score, e.g., F1-score for classification task, V-Measure score for clustering task or visualization quality metric for DR task. The goal is to jointy tune the ensemble of hyperparameters to make the model output the highest score. Trial-and-error method is typically used to test several common combinations of the parameters but it is not a systematic way to tune the hyperparameters.

One common approach to solve this problem is naive grid search. By making a list of discrete values for each hyperparameter, we can try to evaluate all possible combinations. The parameter space growns exponentially w.r.t. the number of hyperparameters and the number of values for each one. A better approach is random search [3], in which we sample randomly the combinations. From the list of alternative values for the hyper-parameters, pick randomly one value for each hyper-parameter to create one combination. But there are some hyper-parameters which have large effect and others which have no effect. If we move along the axes of the no-effect hyperparameters, we do not learn at all. Thus the question is how to jointly tune many hyperparameters at the same time with as few evaluations as possible. Bayesian optimization (BayOpt) is an anwser to our question. We first explain the BayOpt approach in general, then explain how it fits to the hyperparameter tuning problem.

BayOpt is a strategy for finding the extremum (minimum or maximum) of an objective function f [20]. The objective function can be any complex non-convex blackbox function which does not have closed-form expression or its derivative is not accessible. Thus finding directly the extremum of this kind of function is impossible. However, we can observe the function values (possibly noisy) for some sampled input values. The goal of BayOpt is not to approximate the unknown objective function f but instead estimate its extremum (generally speaking, its maximum) from the ensemble of observations in form of pair of input sample x and function values $f(x)$.

Let define $f(x_i)$ as the observation of the target function for the i^{th} sample x_i. BayOpt constructs a statistical model describing the relationship between the tuning hypyerparamters and the target function.

NEED REVIEW: Bayesian model-based optimization is intuitive: choose the next input values to evaluate based on the past results to concentrate the search on more promising values.

BayOpt has been applied successfully to the problem of hyperparameters tuning [24] or experimental design / randomized experiments [17]. We can consider the unknown objective function is our target function (score function) for all combinations of the hyperparameters. Since we can not evaluate all combinations of the hyperparameters, the target function is obviously unknown. The goal is thus to find a best combination that maximize the target function.

TODO:

Discuss: BayOpt can be applied to the BIC-based score [5]. However this score has two disavantages: (1) it is tied to the loss function of t-SNE. (2) it works only with one hyperparameter (perplexity of t-SNE), and thus can not be generalized for other DR methods. Our proposed method can do better.

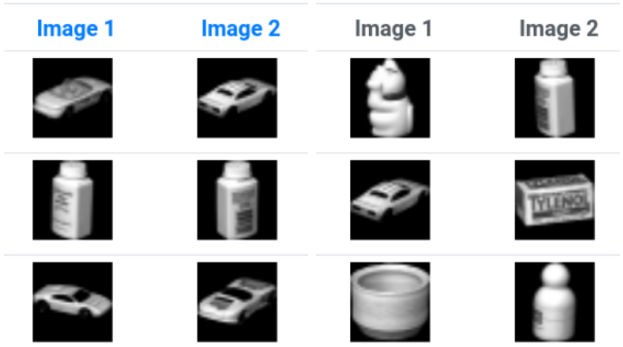
3. Constraint Preserving Score

3.1. Visual Definition of User Pairwise Constraints

Since humans are capable of well distinguishing similar and dissimilar high-dimensional objects with prior knowledge (e.g. comparing countries only by their name, compare images by the visual features such as the shape, color, objects on image, etc.), they can naturally express their cognitive requirements by defining several links between instances. In Fig. 2, several examples are provided of similar-links that can be formed between objects between very similar objects under different point of view while the dissimilar-links would be formed between objects of different shapes.

Our method is based on the following hypothesis: an embedding is said to be good if it accurately represents the high-dimensional data and satisfies the user constraints. The pairwise constraints can then be used as a criterion to fit the user requirements. Our method frees users from manually selecting the hyperparameter of t-SNE and provides a good quality embedding while preserving their predefined constraints. To prove the reliability of this hypothesis, we transform the pairwise constraints to a quantitative score (defined in Section 3.2) and compare it with several embedding quality metrics (presented in Section **??**).

TODO:



(a) Similar-link constraints. (b) Dissimilar-link constraints.

Figure 2: Examples of similar-link and dissimilar-link constraints between pairs of images from COIL20 dataset (a dataset of 20 object captures under many different vantage points).

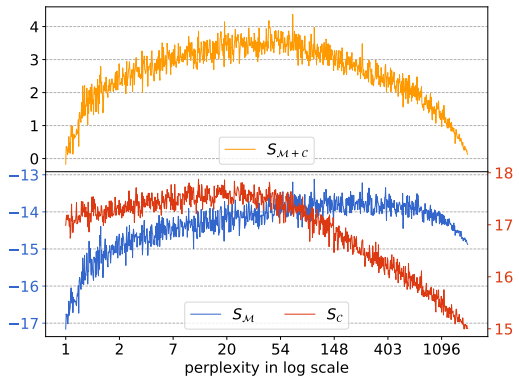


Figure 3: [TODO: Replace]Evolution of constraint-preserving scores $S_{\mathcal{M}+\mathcal{C}}$, $S_{\mathcal{M}}$ and $S_{\mathcal{C}}$ with 50 constraints for *DIGITS* over different perplexities.

+ Explain that we use the auto-generated pairwise constraints when having labels.

+ We can ask the user to label a small proportion of labels in order to construct the pairwise constraints, or the user can select the constraints manually.

3.2. Quantifying the User Constraints

Given an embedding result and a set of user pairwise constraints, our method outputs a score measuring how well the pairwise constraints are preserved in the output embedding, called *constraint-preserving score* S.

Constraint measurement

t-SNE uses the Student's t-distribution to encode the neighborhood information of the instances in the low dimensional space (see Eq. **??**). We also use a t-distribution in the embedded space to quantify the user constraints preservation. Let us denote the embedding result of t-SNE $Y = \{y_i\}$, a set of similar-links $\mathcal{M}$ and a set of dissimilar-links $\mathcal{C}$. The number of similar-link and dissimilar-link constraints are denoted by $|\mathcal{M}|$ and $|\mathcal{C}|$, respectively. For a constrained pair (i, j), the probability of i and j being neighbors is defined as

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

The points connected by a similar-link constraint are considered as neighbors and should be close together. The constrained points in a dissimilar-link should stay apart since they can not be neighbors of each other. Therefore, for each similar-link $(i, j) \in \mathcal{M}$, q_{ij} should be high and inversely, q_{ij} is expected to be low for each dissimilar-link $(i, j) \in \mathcal{C}$.

Constraint-preserving score for similar-links

The amount of similar-link information preserved in a given embedding is measured as a log-likelihood of the joint distribution of q_{ij} over all similar-links $(i, j) \in \mathcal{M}$:

$$S_{\mathcal{M}} = \frac{1}{|\mathcal{M}|} \log \prod_{(i,j)\in\mathcal{M}} q_{ij} = \frac{1}{|\mathcal{M}|} \sum_{(i,j)\in\mathcal{M}} \log q_{ij}. \quad (3)$$

If all pairs connected by similar-links are placed close together, the log-likelihood is high and $S_{\mathcal{M}}$ is maximized.

Constraint-preserving score for dissimilar-links

In contrast to similar-links, the probability q_{ij} for each dissimilar-link $(i, j) \in \mathcal{C}$ should be low, i.e. the log-likelihood over all dissimilar-link pairs ($\log \prod_{\mathcal{C}} q_{ij}$) has to be minimized. Or in other words, the negative log-likelihood over all dissimilar-link constraints should be maximized. The constraint-preserving score for a set of dissimilar-links $\mathcal{C}$ is defined as

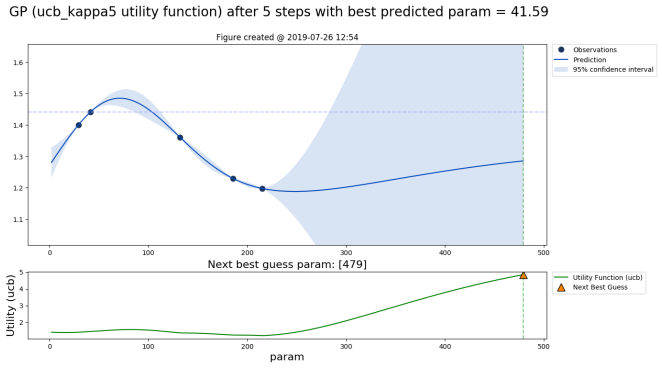
$$S_{\mathcal{C}} = -\frac{1}{|\mathcal{C}|} \log \prod_{(i,j)\in\mathcal{C}} q_{ij} = -\frac{1}{|\mathcal{C}|} \sum_{(i,j)\in\mathcal{C}} \log q_{ij}. \quad (4)$$

By maximizing $S_{\mathcal{C}}$, the embedding that respects the best the dissimilar-link constraints is found. Another way to measure how well a dissimilar-link (i, j) is preserved is to use $1 - q_{ij}$. However, in practice, the value of q_{ij} is very small, which means that $1 - q_{ij}$ is very close to one which makes the log-likelihood of all dissimilar-links vanish.

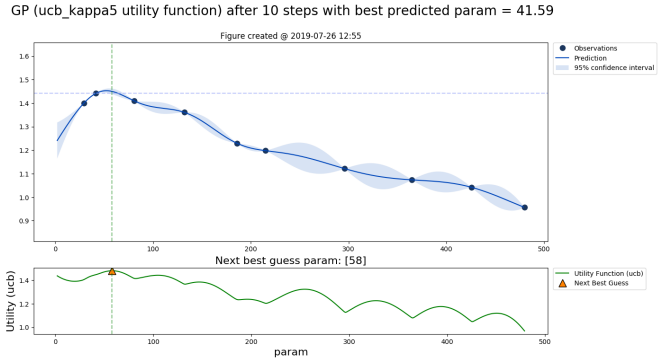
Constraint-preserving score for similar-links and dissimilar-links

The score over all similar-links and dissimilar-links can be maximized at the same time by defining the constraint-preserving score as a combination with equal contribution of both $S_{\mathcal{M}}$ and $S_{\mathcal{C}}$, i.e. $S_{\mathcal{M}+\mathcal{C}} = S_{\mathcal{M}} + S_{\mathcal{C}}$.

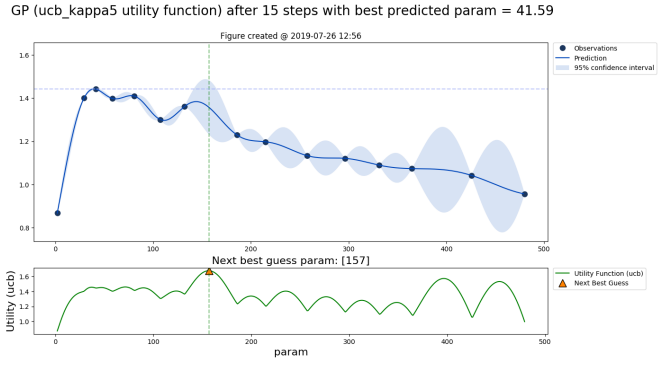
An embedding that retains as much as possible the constraint information is considered to have a good quality with respect to the user needs. Based on these definitions, our method takes a set of user-defined pairwise constraints and searches among all embeddings created by different perplexities for one with maximal $S_{\mathcal{M}+\mathcal{C}}$. Fig. 3 illustrates the contribution of $S_{\mathcal{M}}$ and $S_{\mathcal{C}}$ to the overall $S_{\mathcal{M}+\mathcal{C}}$ score for *DIGITS* dataset with 50 auto-generated constraints (see Section **??** for details about the generation process).



(a) BayOpt after 5 steps.



(b) BayOpt after 10 steps.



(c) BayOpt after 15 steps.

Figure 4: BayOpt in action (COIL20, tsne, ucb, kappa5)

4. Constraint-based Score as Target in Bayesian Optimization Approach

+ Explain the internal step in BayOpt. Can make use some figures Fig. 4a, Fig.4c.

+ Explain how the utility function are constructed and optimized, and answer why optimize the utility function (surrogate function), we can optimize at the same time the target function (the constraint-based score function).

+ Explain the exploitation-exploration trade-off in BayOpt (and estimate the number of times we need to try before reaching to the global maximum).

+ Show that BayOpt gives stable prediction (at least, the best range). Demo with e.g., COIL20 or DIGITS, keep the same input constraint set, change random seed for BayOpt, and get around the same prediction for the best range.

5. Experimental Results

Present the dataset, the used pairwise constraint, the workflow as in the section 5.1, 5.2.

To present and analyze the results, we can present following points (but we still miss the evaluation).

(1) Optimal hyperparameters found by BayOpt w.r.t the constraint-based score:

+ BayOpt method for finding `perplexity` param for t-SNE. (Case of 1 param to tune with t-SNE, Fig. 5) (Done)

+ Add experiment with LargeVis. (TODO)

+ ~~BayOpt for finding n-neighbors param for UMAP. (Case of 1 param to tune with UMAP) (Done)~~

+ BayOpt for finding `n_neighbors` and `min_dist` params for UMAP. (Case of 2 params to tune with UMAP, Fig. 8) (Plan to do)

(2) Visualization of the violated constraints. (Doing, a draft version looks like Fig.9) We can analyze the explainability of the score / the visual assessment of the quality.

(3) Analyse the characteristics of the constraint-based score.

+ Keep the section 6.2:
  - Must-link score agrees with CCA score (Fig 10).
  - Cannot-link score agrees with BIC-based score (Fig. 11).

  - ML+CL agrees with AUC_RNX score (Fig. 12).

+ Variance / Stability of the score: analyze Fig. 13 and explain how BayOpt approach can take into account the variance (uncertainty) of the score.

To discuss:

+ How to evaluate the visualization corresponding to the optimal hyperparam found by our method? => We can show this optimal viz and other 'non-optimal' viz and analyze them visually/intuitively.

+ ~~Compare our score with John's score? (Apply BayOpt to John's metric, compare the predictive score function - the prediction line in the graphs, compare the optimal hyperparameter).~~ (John's metric use HD data, time complexity of $O(DN^3)$ while our score is $O(DN^2)$ with a scale factor of the number of constraints.)

+ Compare our result with the method of auto-selecting perplexity of Cao and Wang? (TODO)

6. Discussion

Meeting 12/07: not sound enough for now, to discuss later.

Discuss the variance of the proposed score (w.r.t to different set of constraints, different number of constraitns). Say, the constraint-based score is a stochastic function. Say,

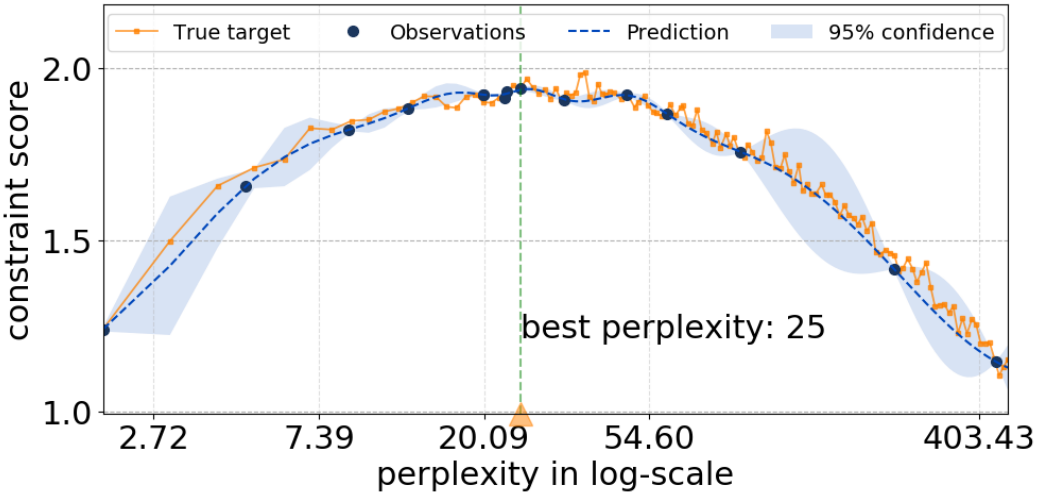


Figure 5: BayOpt with t-SNE with 1 param.

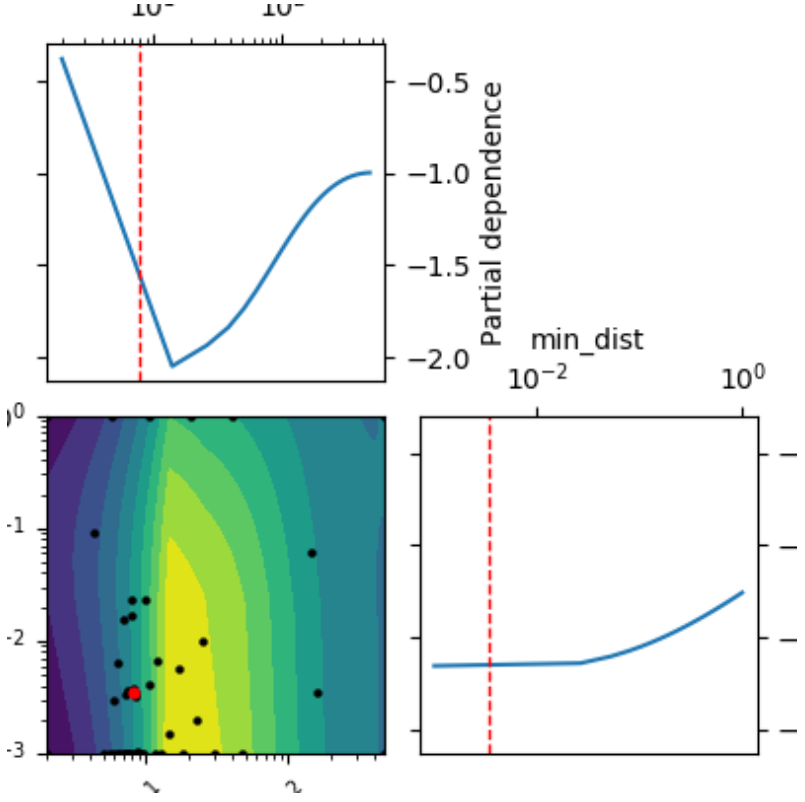


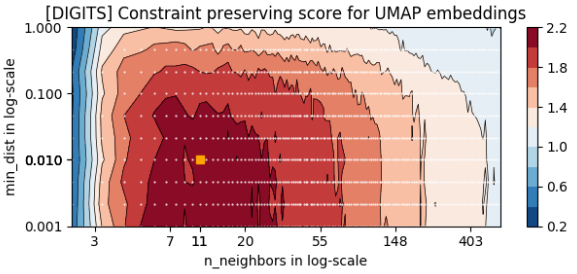
Figure 6: TODO: Produce a figure similar to this one with 2 params of umap.

how BayOpt can take into account the uncertainty (the variance of the score) to estimate the maximum of this stochastic function.

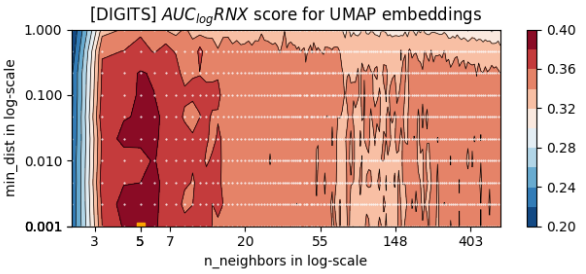
The violated constraints in our methods correspond to the shortcomming of t-SNE in [31]. t-SNE can not preserve the within-cluster distances and between-clusters distances. [TODO: Add reproduced figures]. The q_{ij}-based score can help to understand the defective of the visualization. (E.g., can compare with google's embedding project, vis the neighbors in HD and LD, but it still hard to understand the model. With our score, it is easier to visualize the violated constraints).

Add discussion for UMAP.

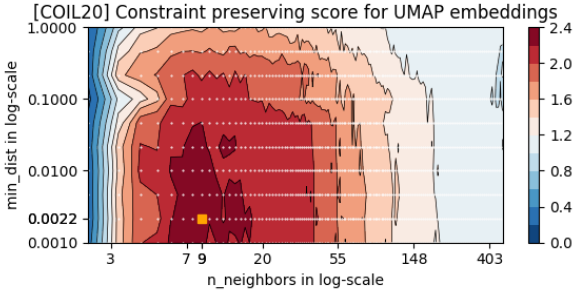
Easily generate pairwise constraints from labels. Only need small amount of labeled points for each class to generate hundreds of constraints. The proposed method only need 200 constraints to work well. [TODO: Test with smaller number of constraints to see if it works. 200 constraints seem



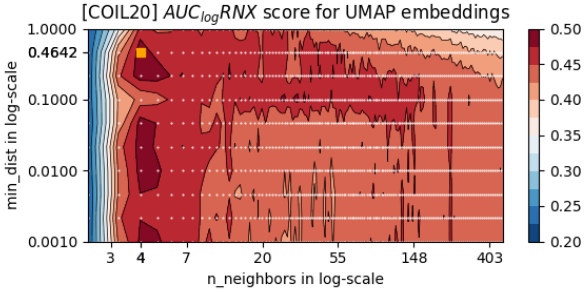
(a) UMAP with DIGITS constraint score



(b) t-SNE with DIGITS John's score



(c) UMAP with COIL20 constraint score



(d) t-SNE with COIL20 John's score

Figure 7: BayOpt with UMAP(nneighbors, mindist)

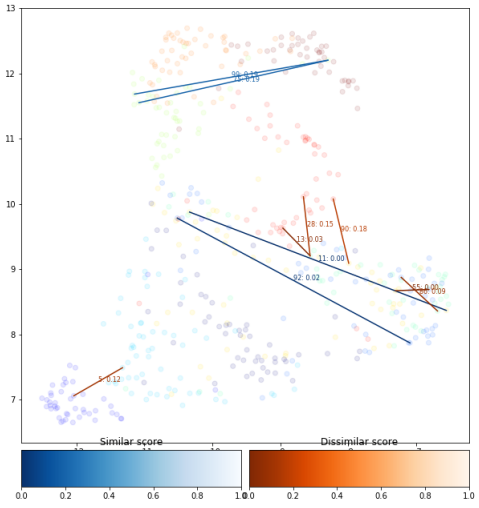


Figure 8: TODO Visualize the constraint score (of the violated links).

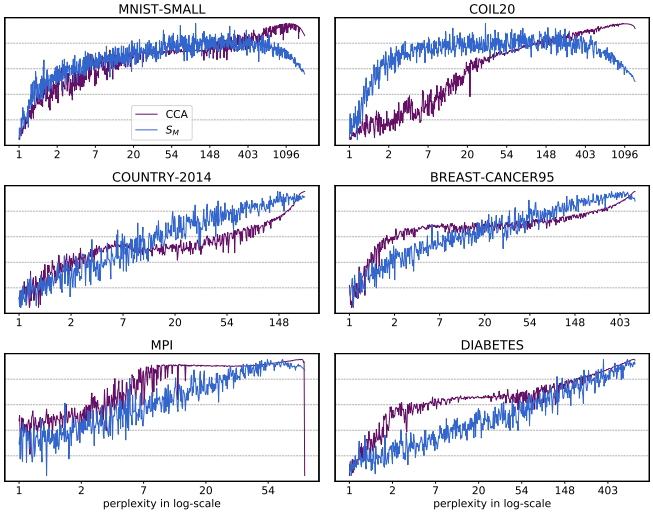


Figure 9: Must-link score agrees with CCA score.

too much].

Can replace the auto-generated constraints by the manual constraints of the real user.

The user can interact directly in the loop of Bayesian Optimization method to select the next hyperparameter to discover.

Both the constraint-based score and the BayOpt's internal steps are explainable even for the non-technical users.

7. Conclusion and Future Work

(1) Repeat the problem of hyperparameter tuning for DR methods and our solution:

+ The proposed constraint-based score is indenpendent to how the embedding is produced and can be used with any DR methods. This score is built upon a limited number of constraints but can distinguish the visualizations prefering local structure and those prefereing global structure.

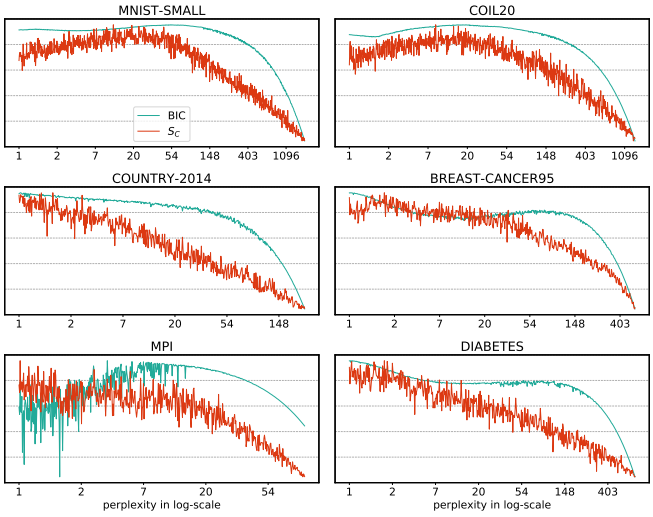


Figure 10: Cannot-link score agrees with BIC-based score.

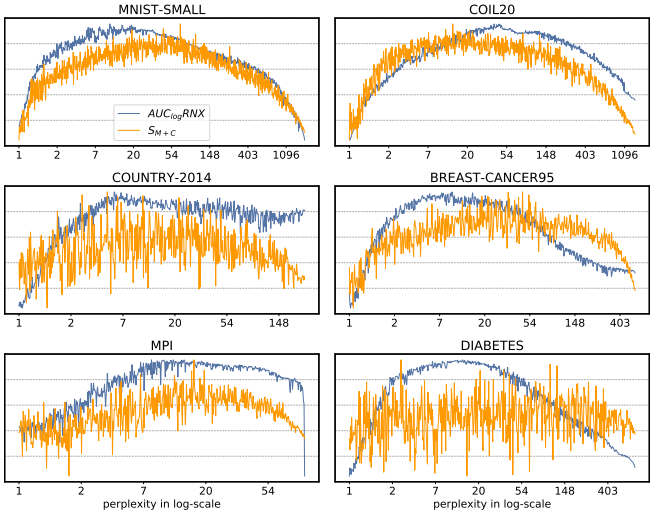


Figure 11: ML+CL agrees with AUC_RNX score.

+ A finding that Bayesian Optimization approach fits well in our problem.

(2) Summary the advantages of the two above elements

+ The constraint-based score agree the the well-known quality metric.

+ This score can be visually represented to explain the violated pairs.

+ By combining this score with BayOpt approach, we can tune many hyperparameters at the same time for many widely used DR methods like t-SNE or UMAP.

+ BayOpt takes into account the uncertainty in the score values and also explainable. We can observe the internal optimization step to anwser the question: why to choose the next promissing hyperparameters to try?

(4) Future work:

(a) User experiment:

+ Integrate the user's feedback in two stages of our workflow. The users can select the pairwise constraints or label some points (used to generate the constraints) to build the score. They can also manualy select the next hyperparameters to evaluate in a customized interactive BayOpt framework.

+ Take the preference of the users on the presented visualizations to evaluate the quality of the visualization. We search for if the best visualization selected by the user corresponds to the result of our method.

(b) Integrate directly the pairwise constraints into the optimization process of BayOpt. BayOpt is now used as a generic toolbox to find the extreme of a blackbox costly objective function. Our idea is to use the pairwise constraint to modify the kernel in the covariance function of Gaussian Process model, which is the core element of BayOpt.

A. Quality metrics

Let d^x_{ij} and d^y_{ij} be, respectively, the distance between instances i and j in HD and LD. Let d^x and d^y be the distances matrices for all pair of points in HD and LD. Here are the mathematical formulas for the five selected metrics.

- The Correlation Coefficient is defined as:

$$\text{CC} = \text{pearson\_correlation}(d^x, d^y) = \frac{\text{Cov}(d^x, d^y)}{\sigma(d^x)\sigma(d^y)}$$

- For measuring the distance order in NMS, an isotonic transformation d^{iso} is performed on d^x. The Kruskal's stress is then computed using this transformation:

$$\text{NMS} = \sqrt{\frac{\sum_{ij}(d^{iso}_{ij} - d^y_{ij})^2}{\sum_{ij} d^y_{ij}}}$$

- The Curvilinear Component Analysis Stress function is defined as:

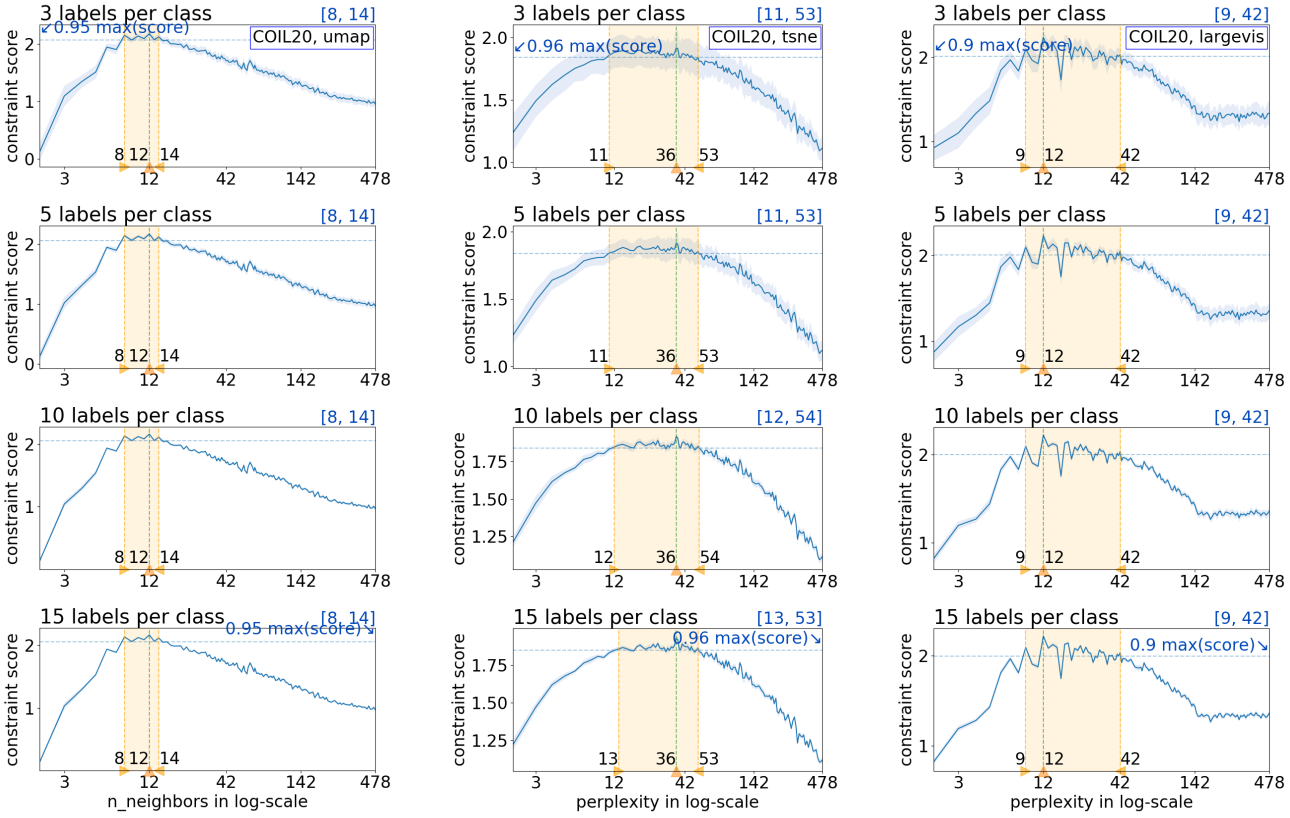
$$\text{CCA} = \sum_{ij}(d^x_{ij} - d^y_{ij})^2 F_\lambda(d^y_{ij}),$$

in which $F_\lambda(d^y_{ij})$ is a decreasing-weighting function of d^y_{ij}. Examples of weighting functions include the step function or $1 - sigmoid(d^y_{ij})$. We used the latter in this paper.

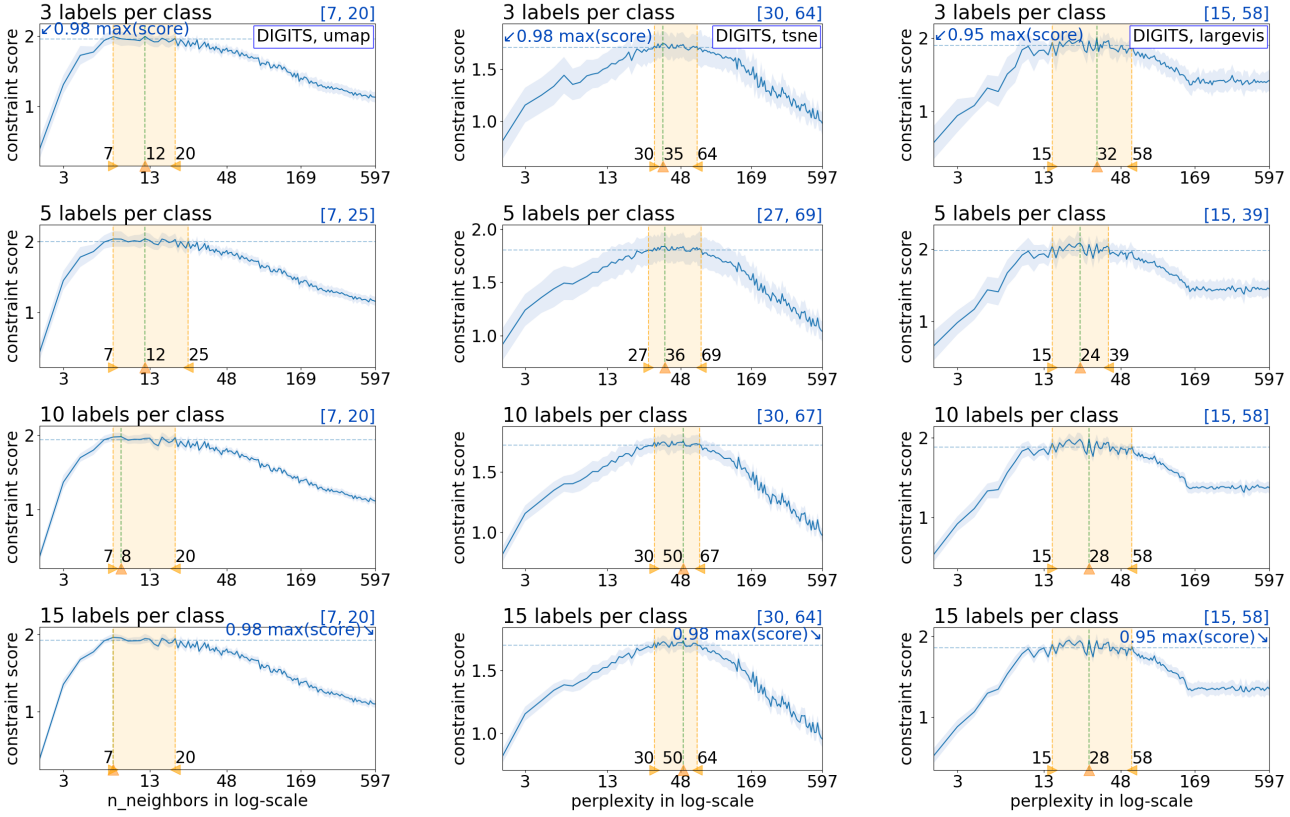
- The stress function of Sammon's Nonlinear mapping is:

$$\text{NLM} = \frac{1}{\sum_{ij} d^x_{ij}} \sum_{ij} \frac{(d^x_{ij} - d^y_{ij})^2}{d^x_{ij}}$$

- The quality measure AUC logRNX can be defined by step:
  - Let k be the number of neighbors considered, N the number of instances, ν^k_i the set of the k closest neighbors of i in the embedding and n^k_i the set



(a) UMAP with COIL20 (b) t-SNE with COIL20 (c) LargeVis with COIL20



(d) UMAP with DIGITS (e) t-SNE with DIGITS (f) LargeVis with DIGITS

Figure 12: Stability of the constraint preserving scores with three methods UMAP, t-SNE and LargeVis for two datasets COIL20 and DIGITS

of the k closest neighbors of i in the HD space, Q_{NX} is defined as:

$$Q_{NX}(k) = \frac{1}{Nk} \sum_{i=1}^{N} |\nu_i^k \cap n_i^k|$$

– $R_{NX}(k)$, the rescaled version of $Q_{NX}(k)$, is defined as:

$$R_{NX}(k) = \frac{(N-1)Q_{NX}(k) - k}{N - 1 - k}$$

– $\text{AUC}_{log}\text{RNX}$ is computed by taking the area under the $R_{NX}(k)$ curve in the log-scale of k:

$$\text{AUC}_{log}\text{RNX} = \left(\sum_{k=1}^{N-2} \frac{R_{NX}(k)}{k} \right) / \left(\sum_{k=1}^{N-2} \frac{1}{k} \right)$$

References

[1] Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D., 2003. Learning distance functions using equivalence relations, in: Proc. 20th Int. Conf. Mach. Learn., pp. 11–18.

[2] Basu, S., Banerjee, A., Mooney, R.J., 2004. Active semi-supervision for pairwise constrained clustering, in: Proc. SIAM Int. Conf. Data Mining, SIAM. pp. 333–344. doi:10.1137/1.9781611972740.31.

[3] Bergstra, J.S., Bardenet, R., Bengio, Y., Kégl, B., 2011. Algorithms for hyper-parameter optimization, in: Advances in neural information processing systems, pp. 2546–2554.

[4] Brochu, E., Cora, V.M., De Freitas, N., 2010. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599 .

[5] Cao, Y., Wang, L., 2017. Automatic selection of t-SNE perplexity, in: ICML AutoML Workshop, Sydney, Australia.

[6] Cevikalp, H., Verbeek, J., Jurie, F., Klaser, A., 2008. Semi-supervised dimensionality reduction using pairwise equivalence constraints, in: VISAPP 3rd Int. Conf. Comput. Vision Theory Appl., pp. 489–496.

[7] Davidson, I., 2009. Knowledge driven dimension reduction for clustering., in: Int. Joint Conf. AI, pp. 1034–1039.

[8] Davidson, I., Basu, S., 2007. A survey of clustering with instance level constraints, in: ACM Trans. Knowl. Discov. Data, pp. 1–41.

[9] Davidson, I., Ravi, S., 2005. Clustering with constraints: Feasibility issues and the k-means algorithm, in: Proc. SIAM Int. Conf. Data Mining, SIAM. pp. 138–149.

[10] Demartines, P., Hérault, J., 1997. Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. IEEE Trans. Neural Netw. 8, 148–154.

[11] Endert, A., et al., 2017. The state of the art in integrating machine learning into visual analytics. Computer Graphics Forum 36, 458–486.

[12] Geng, X., Zhan, D.C., Zhou, Z.H., 2005. Supervised nonlinear dimensionality reduction for visualization and classification. IEEE Trans. Syst., Man, Cybern., Syst, Part B (Cybernetics) 35, 1098–1107.

[13] Giordani, P., Kiers, H.A., 2007. Principal component analysis with boundary constraints. J. Chemometr. 21, 547–556.

[14] Kruskal, J.B., 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika 29, 1–27.

[15] Lee, J.A., Peluffo-Ordóñez, D.H., Verleysen, M., 2014. Multiscale stochastic neighbor embedding: Towards parameter-free dimensionality reduction, in: Proc. 22nd ESANN, Bruges, Belgium. pp. 177–182.

[16] Lee, J.A., Peluffo-Ordóñez, D.H., Verleysen, M., 2015. Multi-scale similarities in stochastic neighbour embedding: Reducing dimensionality while preserving both local and global structure. Neurocomputing 169, 246–261.

[17] Letham, B., Karrer, B., Ottoni, G., Bakshy, E., et al., 2019. Constrained bayesian optimization with noisy experiments. Bayesian Analysis 14, 495–519.

[18] van der Maaten, L., Hinton, G., 2008. Visualizing data using t-sne. J. Mach. Learn. Res. 9, 2579–2605.

[19] McInnes, L., Healy, J., Melville, J., 2018. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426 .

[20] Močkus, J., 1975. On bayesian methods for seeking the extremum, in: Marchuk, G.I. (Ed.), Optimization Techniques IFIP Technical Conference Novosibirsk, July 1–7, 1974, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 400–404.

[21] Mockus, J., Tiesis, V., Zilinskas, A., 1978. The application of bayesian methods for seeking the extremum. Towards global optimization 2, 2.

[22] Sacha, D., et al., 2017. Visual interaction with dimensionality reduction: A structured literature analysis. IEEE Trans. Vis. Comput. Graphics 23, 241–250.

[23] Sammon, J.W., 1969. A nonlinear mapping for data structure analysis. IEEE Trans. Comput. 18, 401–409.

[24] Snoek, J., Larochelle, H., Adams, R.P., 2012. Practical bayesian optimization of machine learning algorithms, in: Advances in neural information processing systems, pp. 2951–2959.

[25] Strickert, M., 2012. No perplexity in stochastic neighbor embedding, in: New Challenges in Neural Computation Workshop, p. 68.

[26] Sugiyama, M., Idé, T., Nakajima, S., Sese, J., 2008. Semi-supervised local fisher discriminant analysis for dimensionality reduction. Machine Learning 78, 35–61. doi:10.1007/s10994-009-5125-7.

[27] Tang, J., Liu, J., Zhang, M., Mei, Q., 2016. Visualizing large-scale and high-dimensional data, in: Proceedings of the 25th international conference on world wide web, International World Wide Web Conferences Steering Committee. pp. 287–297.

[28] Tang, W., Zhong, S., 2007. Pairwise constraints-guided dimensionality reduction, in: Computational Methods of Feature Selection. Chapman & Hall, pp. 295–312.

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

[30] Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S.e.a., 2001. Constrained k-means clustering with background knowledge, in: Proc. 18th Int. Conf. Mach. Learn., pp. 577–584.

[31] Wattenberg, M., Viégas, F., Johnson, I., 2016. How to use t-sne effectively. Distill 1, e2.

[32] Xing, E.P., Jordan, M.I., Russell, S.J., Ng, A.Y., 2003. Distance metric learning with application to clustering with side-information, in: Adv. in Neur. Inf. Process. Syst. 15, pp. 521–528.

[33] Zhang, D., Zhou, Z.H., Chen, S., 2007. Semi-supervised dimensionality reduction, in: Proc. SIAM Int. Conf. Data Mining, SIAM. pp. 629–634.