Counterfactual metrics*

André Artelt[1] and Barbara Hammer[1]

CITEC - Cognitive Interaction Technology
Bielefeld University, 33619 Bielefeld, Germany
{aartelt,bhammer}@techfak.uni-bielefeld.de

Abstract. One popular explanation scheme for decisions of machine learning models is offered by counterfactuals, i.e. an explication of which change of a given input would lead to a different class label. In this contribution, we take a different stance and consider the question, given a model and novel data points with a desired classification, which minimum change to the model would be necessary to classify the new data correctly, aiming for an explanation of expected changes if a model is retrained or, within a lifelong learning scheme, continuously adapted to new data. We focus on distance-based classifiers, where model changes correspond to changes of metric parameters, and we develop efficient optimization techniques to generate such counterfactual metric changes depending on the chosen model.

Keywords: Counterfactual Explanations · XAI · Metric Learning · Learning with Drift

1 Introduction

The increasing popularity of machine learning models in the consumer domain, in industrial applications, or in information processing and social media led to the demand for transparency of suggestions or decisions induced by such models. Various approaches deal with the question how to accompany machine learning models by aspects which can explain their functionality, see e.g. [8, 14] for recent overviews and [1, 2] for first toolboxes how to generate explanations of machine learning models. Most explanation techniques aim for an explanation of either an existing model as a whole, or an explanation of a specific decision proposed by a model. In the latter situation, counterfactual explanations as proposed by [18] constitute one popular tool: given an input, a counterfactual corresponds to an alternative data point, which is similar to the given data point but classified differently, i.e. a counterfactual suggests how to minimally change the input when aiming for a different decision. Counterfactuals can be computed agnostically, yet model specific optimization schemes such as proposed in [4] can lead to an enhanced performance and avoid ambiguities in some cases. Further there do

* We gratefully acknowledge funding from the VW-Foundation for project *IMPACT* funded in the frame of the funding line *AI and its Implications for Future Society.*

exist approaches to provide only those counterfactuals which are actionable as regards the structure of the underlying data manifold [12].

In this work, we take a different stance: given a data point and a model decision, we address the question how to minimally change the model rather than the input data to provide a desired outcome. This question is relevant as soon as the objective is the explanation of changes of a model if it would be adapted to novel data – a situation which occurs when an existing model should to be retrained based on new observations in scenarios with periodical updates or continuous learning; explanations of how the model changes based on the new data are relevant for the decision whether retraining is necessary and which behavior can be expected after model adaptation.

Explanation of model changes depend on the model functionality and its expected change, on the one hand, and a suitable choice of a vocabulary to express such changes. Obviously, one way to express model change is in terms of typical data points for which the model functionality changes. Here, we are interested in a different way to explain the model change, namely the relevance of features for a model to take decisions; note that this language is also used to express explanations of model decisions in popular approaches such as LIME [13]. The question of feature relevance changes could be addressed agnostically, relying e.g. on popular measures as proposed in the context of feature selection such as mutual information. Here, we are interested only in a specific type of model, for which feature relevance changes come as a particularly natural explanation: metric- or distance-based classification schemes. Many popular machine learning (ML) models like kNN, RBF-networks and LVQ [11] are based on distances. While often the standard Euclidean distance is used as a default metric, in (distance) metric learning schemes [6,9] aim for an estimation of an optimum Mahalanobis (distance) metric from a given training data set [9].

Here we will focus on two popular supervised distance-based classifiers: prototype-based classifiers with metric learning [15,16] and metric learning for k-nearest neighbor approaches [19]. For these models, counterfactual metric learning corresponds to the problem how to minimally change the Mahalanobis distance given novel training data. In this contribution, we will phrase this problem in terms of an optimization problem, which, depending on the specific type of model, is convex and can be solved efficiently. We demonstrate the performance of the scheme in an indirect way, since a direct evaluation would require user studies in a specific domain: we evaluate the performance of the model by evaluating its availability to characterize underlying known metric drift based on observed data.

2 Foundations

Learning vector quantization: All (supervised) learning vector quantization (LVQ) models [11] compute a set of labeled prototypes $\{(\mathbf{p}_i, o_i)\}$ from a given training data set - we refer to the i-th prototype as $\mathbf{p}_i$ and the corresponding label as o_i.

The prediction function h of a LVQ model is then given as:

$$\begin{aligned} & h(\boldsymbol{x}) = o_i \\ & \text{s.t. } \min \mathrm{d}(\boldsymbol{x}, \mathbf{p}_i) \end{aligned} \tag{1}$$

where $\mathrm{d}(\cdot)$ refers to a distance function. In vanillas LVQ, the squared Euclidean distance $\mathrm{d}(\boldsymbol{x}, \mathbf{p}_i) = (\boldsymbol{x} - \mathbf{p})^\top \mathbb{I}(\boldsymbol{x} - \mathbf{p})$ is used. Matrix-LVQ (e.g. GMLVQ [15]) replaces the identity matrix $\mathbb{I}$ by a s.psd distance matrix that is estimated from the data, and thus gives rise to the Mahalanobis distance:

$$\mathrm{d}_{\boldsymbol{\Omega}}(\boldsymbol{x}, \mathbf{p}) = (\boldsymbol{x} - \mathbf{p})^\top \boldsymbol{\Omega}(\boldsymbol{x} - \mathbf{p}) \tag{2}$$

where $\boldsymbol{\Omega}$ denotes the s.psd distance matrix that is estimated from the data.

Instead of a single global distance matrix, local-matrix LVQ (e.g. LGMLVQ [16]) uses prototype (or class) specific distance matrices $\boldsymbol{\Omega}_i$. Thus the Mahalanobis distance between a point $\boldsymbol{x}$ and a prototype $\mathbf{p}_i$ becomes:

$$\mathrm{d}_{\boldsymbol{\Omega}_i}(\boldsymbol{x}, \mathbf{p}_i) = (\boldsymbol{x} - \mathbf{p}_i)^\top \boldsymbol{\Omega}_i(\boldsymbol{x} - \mathbf{p}_i) \tag{3}$$

More details on the estimation of the parameters $\{(\mathbf{p}_i, o_i)\}$ and $\boldsymbol{\Omega}_i$ can be found in [15,16].

Counterfactual explanations: When it comes to applying & deploying ML models in the real world, interpertability and explainability become important [17]. Counterfactual explanations [18] (often just called counterfactuals) are a popular strategy for explaining [17] the prediction of a ML model. Counterfactuals are considered to be very human-friendly and useful [10,18]. A counterfactual states a minimal change to the input that lead to a different (requested) behaviour of the model. For a given prediction function $h : \mathcal{X} \mapsto \mathcal{Y}$ and input $\boldsymbol{x} \in \mathbb{R}^d$, the finding of a counterfactual $\boldsymbol{x}' \in \mathbb{R}^d$ can be formalized as the following optimization problem [18]

$$\underset{\boldsymbol{x}' \,\in\, \mathbb{R}^d}{\arg\min}\; \ell\left(h(\boldsymbol{x}'), y'\right) + C \cdot \theta(\boldsymbol{x}', \boldsymbol{x}) \tag{4}$$

where $\ell(\cdot)$ denotes a loss function that penalizes deviation of the prediction $h(\boldsymbol{x}')$ from the requested prediction y'. $\theta(\cdot)$ denotes a regularization that penalizes deviations from the original input $\boldsymbol{x}$, and the hyperparameter C denotes the regularization strength. Two common regularizations [18] are the weighted Manhattan distance and the Mahalanobis distance.

An comprehensive overview on the computation of counterfactual explanations of different ML models is given in [5].

Instead of changing the data point - like it is done in counterfactual explanations -, we propose to change the distance metric of the model. We explain the prediction of a distance based model on a given data point, by stating a minimal change to the distance metric that lead to a different (specified) prediction on this data point. We think that knowing how to change the model in order to get a requested prediction provides us with additional insights on the inner working of the model.

More precisely, our contributions are:

- We propose a new method - called *counterfactual metrics* - for explaining the prediction of distance based ML models. We develop convex programs for computing counterfactual metrics of specific ML models.
- We propose to use counterfactual metrics for adapting specific distance based models to a drifting metric. Furthermore, we use counterfactual metrics for explaining the drift.

3 Counterfactual metrics

Under the assumption that the distance based prediction function $h : \mathcal{X} \mapsto \mathcal{Y}$ depends on a distance matrix $\boldsymbol{\Omega} \in \mathcal{S}_{+}^{d}$ - e.g. it uses the Mahalanobis distance - we write $h_{\boldsymbol{\Omega}}(\boldsymbol{x})$, we formalize the procedure for computing a *counterfactual metric* $\boldsymbol{\Omega}' \in \mathcal{S}_{+}^{d}$ as the following optimization problem:

$$\underset{\boldsymbol{\Omega}' \in \mathcal{S}_{+}^{d}}{\arg\min}\; \theta(\boldsymbol{\Omega}', \boldsymbol{\Omega}) \tag{5a}$$

$$\text{s.t. } h_{\boldsymbol{\Omega}'}(\boldsymbol{x}) = y' \tag{5b}$$

where $\mathcal{S}_{+}^{d}$ denotes the set of s.psd matrices in $\mathbb{R}^{d\times d}$ and $\theta(\boldsymbol{\Omega}', \boldsymbol{\Omega})$ penalizes the difference between the counterfactual distance matrix $\boldsymbol{\Omega}'$ and the original distance matrix $\boldsymbol{\Omega}$. There exist many functions that are reasonable choices for $\theta(\cdot)$. A possible choice is the squared Frobenius norm[1] given as:

$$\theta(\boldsymbol{\Omega}', \boldsymbol{\Omega}) = \|\boldsymbol{\Omega}' - \boldsymbol{\Omega}\|_F^2 = \sum_{i,j} (\boldsymbol{\Omega}' - \boldsymbol{\Omega})_{i,j}^2 \tag{6}$$

or the ℓ_1 matrix norm given as:

$$\theta(\boldsymbol{\Omega}', \boldsymbol{\Omega}) = \|\boldsymbol{\Omega}' - \boldsymbol{\Omega}\|_1 = \sum_{i,j} |(\boldsymbol{\Omega}' - \boldsymbol{\Omega})_{i,j}| \tag{7}$$

We can interpret the diagonal of $|\,\boldsymbol{\Omega} - \boldsymbol{\Omega}'\,|$[2] as the individual feature importances of the given data point $\boldsymbol{x}$. *How do we have to change the importance of the features so that the model predicts differently on the given data point - which features caused the original prediction of the model?*

4 Computation of counterfactual metrics

4.1 The general case

In general, we can solve the optimization problem Eq. (5) by using an iterative solver. If Eq. (5) is differentiable with respect to $\boldsymbol{\Omega}'$, we can use gradient based methods. However, because we have to ensure that $\boldsymbol{\Omega}'$ is a s.psd matrix, it

[1] also called matrix ℓ_2 norm

[2] $|\cdot|$ denotes the matrix of the absolute element wise differences.

might be beneficial to decompose[3] $\mathbf{\Omega} = \mathbf{L}^\top\mathbf{L}$ such that we only have to estimate a real matrix $\mathbf{L}$ without any additional constraints on the definiteness of $\mathbf{L}$. Furthermore, in order to reduce the computational complexity of the task, we might reduce the rank of $\mathbf{L}$ so that we have to estimate fewer parameters.

On the other hand, if Eq. (5) is not differentiable with respect to $\mathbf{\Omega}$, we have to use a black-box solver like the Downhill-Simplex method.

Note that, depending on the model $h(\cdot)$ and the regularization $\theta(\cdot)$, the optimization problem Eq. (5) might have local (non-global) optima.

In the next section we investigate - for the special cases of specific LVQ models - how to solve the optimization problem Eq. (5) efficiently.

4.2 Global distance matrix

Because a LVQ model assigns the label of the nearest prototype to a given input, the nearest prototype must be a prototype $\mathbf{p}_i$ with $o_i = y'$. Similar to [3], for computing a counterfactual distance matrix $\mathbf{\Omega}$' it is sufficient to solve the following optimization problem for each prototype $\mathbf{p}_i$ with $o_i = y'$ and select the counterfactual distance matrix $\mathbf{\Omega}$' that yields the smallest value of $\theta(\mathbf{\Omega}, \mathbf{\Omega}')$:

$$\begin{aligned}&\underset{\mathbf{\Omega}\text{'} \in \mathcal{S}^d_+}{\arg\min}\ \theta(\mathbf{\Omega}\text{'}, \mathbf{\Omega}) \\ &\text{s.t. } \mathrm{d}_{\mathbf{\Omega}\text{'}}(\boldsymbol{x}, \mathbf{p}_i) + \epsilon \leq \mathrm{d}_{\mathbf{\Omega}\text{'}}(\boldsymbol{x}, \mathbf{p}_j) \quad \forall\, \mathbf{p}_j \in \mathcal{P}(y')\end{aligned} \tag{8}$$

where $\mathcal{P}(y')$ denotes the set of all prototypes <u>not</u> labeld as y' and $\epsilon > 0$ is a small number making sure that the data point does not lie on the decision boundary.

We can interpret linear discriminant analysis (LDA) as a special case of GMLVQ (although the training is quite different) where each class is represented by exactly one prototype. Therefore, we can use Eq. (8) for LDA models, too. In case of GMLVQ, the constraint in Eq. (8) can be written as:

$$\boldsymbol{x}^\top \mathbf{\Omega}\text{'}(\mathbf{p}_j - \mathbf{p}_i) + \frac{1}{2}\left(\mathbf{p}_i^\top \mathbf{\Omega}\text{'}\,\mathbf{p}_i - \mathbf{p}_j^\top \mathbf{\Omega}\text{'}\,\mathbf{p}_j\right) + \epsilon \leq 0 \quad \forall\, \mathbf{p}_j \in \mathcal{P}(y') \tag{9}$$

If we use the squared Frobenius norm (Eq. (6)) or the matrix ℓ_1 norm (Eq. (7)) as a regularizer $\theta(\cdot)$, the optimization problem Eq. (8) can be rewritten[4] as a semi definite-program (SDP). Note that SDPs can be solved efficiently [7].

4.3 Local distance matrices

In case of local distance matrices (prototype or class wise), the optimization problem Eq. (8) becomes:

$$\begin{aligned}&\underset{\{\mathbf{\Omega}\text{'}_i \in \mathcal{S}^d_+\}}{\arg\min}\ \sum_i \theta(\mathbf{\Omega}\text{'}_i, \mathbf{\Omega}_i) \\ &\text{s.t. } \mathrm{d}_{\mathbf{\Omega}\text{'}_i}(\boldsymbol{x}, \mathbf{p}_i) + \epsilon \leq \mathrm{d}_{\mathbf{\Omega}\text{'}_j}(\boldsymbol{x}, \mathbf{p}_j) \quad \forall\, \mathbf{p}_j \in \mathcal{P}(y')\end{aligned} \tag{10}$$

[3] Because $\mathbf{\Omega}$' is a symmetric positive definite matrix, we know that such a decomposition exists - e.g. Cholesky decomposition.

[4] Note that all derivations can be found in appendix A.

Note that we optimize over a set of local distance matrices simultaneously. In case of LGMLVQ, the constraint in Eq. (10) can be written as:

$$\frac{1}{2}\boldsymbol{x}^\top \left(\boldsymbol{\Omega}'_i - \boldsymbol{\Omega}'_j\right) \boldsymbol{x} + \boldsymbol{x}^\top \left(\boldsymbol{\Omega}'_j \mathbf{p}_j - \boldsymbol{\Omega}'_i \mathbf{p}_i\right) + \frac{1}{2}\left(\mathbf{p}_i^\top \boldsymbol{\Omega}'_i \mathbf{p}_i - \mathbf{p}_j^\top \boldsymbol{\Omega}'_j \mathbf{p}_j\right) + \epsilon \leq 0 \;\forall\, \mathbf{p}_j \in \mathcal{P}(y') \tag{11}$$

Similar to the global distance matrix case, the optimization problem Eq. (10) is a semi-definite program (SDP) which can be solved efficiently, if we use the squared Frobenius norm (Eq. (6)) or the matrix ℓ_1 norm (Eq. (7)) as a regularizer $\theta(\cdot)$.

However, unlike in case of LDA, quadratic discriminant analysis (QDA) does not yield the same optimization problem as LGMLVQ does. This is because of the class dependent normalizing factor which depends on the distance matrix $\boldsymbol{\Omega}_i$ - actually, QDA yields a non-convex problem.

5 Application: Explaining and adapting to a drifting metric

We consider a scenario, where we have a generating process that generates samples from a multi-class classification problem. We assume that the samples are drawn from a class dependent multivariate normal distribution. Furthermore, we assume that the mean of these distributions is fixed but the covariance matrices are changing over time.

We assume that we are given full access to a fitted matrix-LVQ model where each class is represented by a single prototype. In addition, we are provided with new labeled data points $\{(\boldsymbol{x}_i, y_i)\}$ as batches (of size ≥ 1) over time. Our goal is to explain the drift/adaptation by using counterfactual metrics and adapt the given model to the drifting covariance matrices of the generating process such that the number of errors is minimized while avoiding catastrophic forgetting.

In case of a GMLVQ (or LDA) model, we model the adaptation as the following optimization problem:

$$\underset{\boldsymbol{\Omega}' \in \mathcal{S}^d_+}{\arg\min}\; \lambda\, \theta(\boldsymbol{\Omega}', \boldsymbol{\Omega}) + (1-\lambda)\sum_i \xi_i \tag{12a}$$

$$\text{s.t. } \mathrm{d}_{\boldsymbol{\Omega}'}(\boldsymbol{x}_i, \mathbf{p}_{y_i}) - \mathrm{d}_{\boldsymbol{\Omega}'}(\boldsymbol{x}_i, \mathbf{p}_j) - \xi_i + \epsilon \leq 0 \quad \forall\, i,\ j \neq y_i \tag{12b}$$

$$\xi_i \geq 0 \quad \forall\, i \tag{12c}$$

where we introduced slack variables ξ_i for taking into account that the classification data set might not be separable by the model. We use λ for controlling the amount of changes vs. errors that are allowed. Note that this optimization problem is equivalent (up to the slack variables ξ_i) to Eq. (5) for computing a counterfactual metric. Hence, we use the solution of Eq. (12) as the adapted distance matrix of the model, and the difference between the old distance matrix and the new one as a counterfactual explanation of the drift.

Similar, in case of a LGMLVQ model, we combine Eq. (12) with the optimization problem for computing a counterfactual of a LGMLVQ model Eq. (10)

as follows:

$$\underset{\{\mathbf{\Omega'}_i \in \mathcal{S}^d_+\}}{\arg\min} \; \lambda \sum_i \theta(\mathbf{\Omega'}_i, \mathbf{\Omega}_i) + (1-\lambda) \sum_i \xi_i \tag{13a}$$

$$\text{s.t. } \mathrm{d}_{\mathbf{\Omega'}_{y_i}}(\boldsymbol{x}_i, \mathbf{p}_{y_i}) - \mathrm{d}_{\mathbf{\Omega'}_j}(\boldsymbol{x}_i, \mathbf{p}_j) - \xi_i + \epsilon \leq 0 \quad \forall\, i,\; j \neq y_i \tag{13b}$$

$$\xi_i \geq 0 \quad \forall\, i \tag{13c}$$

In case of GMLVQ (or LDA), the constraint Eq. (12b) becomes:

$$\boldsymbol{x}_i^\top \left(\mathbf{\Omega'}(\mathbf{p}_j - \mathbf{p}_{y_i})\right) + \frac{1}{2}\left(\mathbf{p}_{y_i}^\top \mathbf{\Omega'} \mathbf{p}_{y_i} - \mathbf{p}_j^\top \mathbf{\Omega'} \mathbf{p}_j\right) - \xi_i + \epsilon \leq 0 \quad \forall\, i,\; j \neq y_i \tag{14}$$

Likewise, in case of LGMLVQ, the constraint Eq. (13b) becomes:

$$\frac{1}{2}\boldsymbol{x}_i^\top \left(\mathbf{\Omega'}_{y_i} - \mathbf{\Omega'}_j\right) \boldsymbol{x} + \boldsymbol{x}^\top \boldsymbol{q}_{ij} + c_{ij} - \xi_i + \epsilon \leq 0 \quad \forall\, i,\; j \neq y_i \tag{15}$$

where

$$\boldsymbol{q}_{ij} = \mathbf{\Omega'}_j \mathbf{p}_j - \mathbf{\Omega'}_{y_i} \mathbf{p}_{y_i} \qquad c_{ij} = \frac{1}{2}\left(\mathbf{p}_{y_i}^\top \mathbf{\Omega'}_{y_i} \mathbf{p}_{y_i} - \mathbf{p}_j^\top \mathbf{\Omega'}_j \mathbf{p}_j\right) \tag{16}$$

In both cases, the constraints can be written[5] as semi-definite constraints. Therefore, the optimization problems Eq. (12) and Eq. (13) can be written as SDPs and thus be solved efficiently [7].

5.1 Experiments

We have a generating process with two classes where the class dependent distributions differ in the mean only. Initially, we are provided with 200 samples from each class which we use for fitting a GMLVQ model (one prototype per class). We then change the covariance matrix of the generating process over time. After each change, we sample 200 samples per class and use this samples to adapt the GMLVQ model to the drift - see Fig. 5 for an illustration. We assess the performance of the adaption by calculating the accuracy on the new samples before and after adapting the model. The accuracy scores over time are shown in Fig. 1. It can be clearly seen, that the

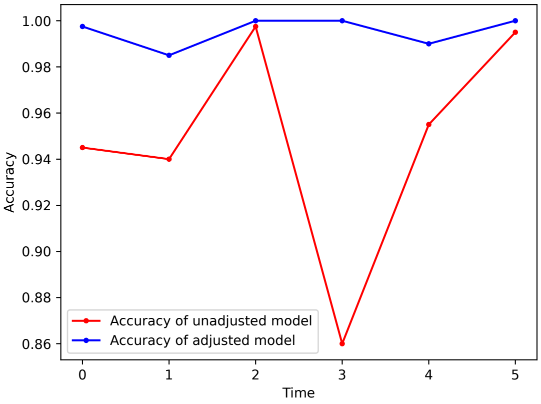

Fig. 1: Accuracy of the adapted and not adapted model over time.

[5] See appendix A for details.

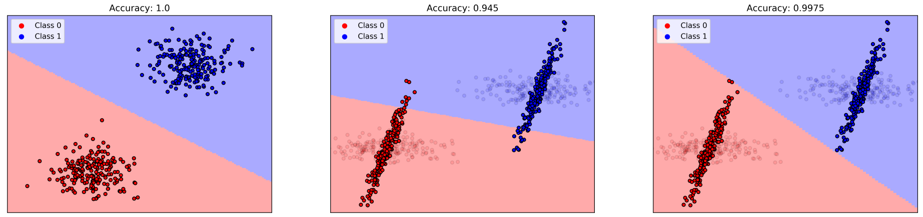

Fig. 2: Original data set and learned decision boundary.

Fig. 3: Drifted samples and the original decision boundary.

Fig. 4: Drifted samples and the adapted decision boundary.

Fig. 5: Adapting to a drifting covariance matrix.

model successfully adapts to the drifting covariance matrix of the generating process. When comparing the ground truth changes of the covariance matrix with the changes in the distance matrix, we observe that in most cases the counterfactual metric successfully identifies the changed correlations - in some cases the counterfactual metric misses some correlations because they are not needed for adapting to the model to the new samples.

5.2 Conclusion

Inspired by counterfactual explanations, we proposed *counterfactual metrics* for explaining distance matrix based models. Instead of changing a given data point for obtaining a requested prediction, we state a minimum change to the distance matrix of the model for getting the requested prediction on the given a data point.

We proposed semi-definite programs (SDPs) for efficiently computing counterfactual metrics of matrix-LVQ models. This modeling allows us to add further constraints (e.g. box-constraints) as well as formal guarantees on the solution (like uniqueness) - in particular, for a given scenario it can be easily checked whether a solution (counterfactual metric) exists or not.

We proposed to use counterfactual metrics for explaining and adapting matrix-LVQ models to a drift metric process. We setup a toy example and empirically demonstrated that our counterfactual metrics reliably identify the drifting components and are able to adapt the model to the drifting metric.

A Appendix

In the subsequent sections we show how to explicitly write the proposed optimization problems as semi definite-programs (SDPs).

A.1 Regularizers

Recall that our objective is given as:

$$\underset{\mathbf{\Omega}\text{'} \,\in\, \mathcal{S}_+^d}{\arg\min}\; \theta(\mathbf{\Omega}\text{'}, \mathbf{\Omega}) \tag{17}$$

Next, we show that Eq. (17) can be written as a SDP - we consider the squared Frobenius norm and the matrix ℓ_1 norm only.

Squared Frobenius norm In case of the squared Frobenius norm (Eq. (6)), we can rewrite Eq. (17) in epigraph form with explicit semi-definite constraints:

$$\begin{aligned} &\min_{\substack{\mathbf{\Omega}\text{'} \,\in\, \mathbb{R}^{d\times d} \\ t \,\in\, \mathbb{R}}} t \\ &\text{s.t. } \; \mathbf{\Omega}\text{'} \succeq 0 \\ &\begin{pmatrix} \mathbb{I} & \boldsymbol{\mathit{\Omega}'} \\ \boldsymbol{\mathit{\Omega}'}^\top & 2\boldsymbol{\mathit{\Omega}'}^\top \boldsymbol{\mathit{\Omega}} + t \end{pmatrix} \succeq 0 \end{aligned} \tag{18}$$

where we made use of the Schur complement and defined $\boldsymbol{\mathit{\Omega}'}$,$\boldsymbol{\mathit{\Omega}}$ to be the flattened version of $\mathbf{\Omega}$',$\mathbf{\Omega}$.

Matrix $\boldsymbol{\ell_1}$ norm In case of the matrix ℓ_1 norm (Eq. (7)), we can rewrite Eq. (17) in epigraph form with explicit semi-definite constraints:

$$\begin{aligned} &\min_{\substack{\mathbf{\Omega}\text{'} \,\in\, \mathbb{R}^{d\times d} \\ t \,\in\, \mathbb{R}}} t \\ &\text{s.t. } \; \operatorname{diag}\left(\boldsymbol{\mathit{\Omega}'} + \boldsymbol{\beta} - \boldsymbol{\mathit{\Omega}}\right) \succeq 0 \quad \operatorname{diag}\left(\boldsymbol{\beta}\right) \succeq 0 \quad \mathbf{\Omega}\text{'} \succeq 0 \\ &\begin{pmatrix} 1 & 0 \\ 0 & -\mathbf{1}^\top \boldsymbol{\beta} + t \end{pmatrix} \succeq 0 \quad \operatorname{diag}\left(\boldsymbol{\mathit{\Omega}} + \boldsymbol{\beta} - \boldsymbol{\mathit{\Omega}'}\right) \succeq 0 \end{aligned} \tag{19}$$

where we again made use of the Schur complement and defined $\boldsymbol{\mathit{\Omega}'}$, $\boldsymbol{\mathit{\Omega}}$ to be the flattened version of $\mathbf{\Omega}$',$\mathbf{\Omega}$ and introduced an auxiliary variable $\boldsymbol{\beta}$.

A.2 Constraints

Recall that we have a bunch of constraints of the following form:

$$\mathrm{d}_{\mathbf{\Omega}\text{'}_i}(\boldsymbol{x}, \mathbf{p}_i) - \mathrm{d}_{\mathbf{\Omega}\text{'}_j}(\boldsymbol{x}, \mathbf{p}_j) + \epsilon \leq 0 \tag{20}$$

In case of a single global distance matrix, we have $\mathbf{\Omega}\text{'}_i = \mathbf{\Omega}\text{'}_j = \mathbf{\Omega}\text{'}$. Then, constraint Eq. (20) can be written as:

$$\boldsymbol{x}^\top \mathbf{\Omega}\text{'}(\mathbf{p}_i - \mathbf{p}_j) - \frac{1}{2}\left(\mathbf{p}_i^\top \mathbf{\Omega}\text{'}\, \mathbf{p}_i - \mathbf{p}_j^\top \mathbf{\Omega}\text{'}\, \mathbf{p}_j\right) - \epsilon \geq 0 \tag{21}$$

In case of local distance matrices, the constraint Eq. (20) can be written as:

$$\frac{1}{2}\boldsymbol{x}^\top\left(\boldsymbol{\Omega}\text{'}_j - \boldsymbol{\Omega}\text{'}_i\right)\boldsymbol{x} - \boldsymbol{x}^\top(\boldsymbol{\Omega}\text{'}_j\mathbf{p}_j - \boldsymbol{\Omega}\text{'}_i\mathbf{p}_i) - \frac{1}{2}\left(\mathbf{p}_i^\top\boldsymbol{\Omega}\text{'}_i\mathbf{p}_i - \mathbf{p}_j^\top\boldsymbol{\Omega}\text{'}_j\mathbf{p}_j\right) - \epsilon \geq 0 \tag{22}$$

Because Eq. (21) and Eq. (22) are both affine in $\boldsymbol{\Omega}$' (or $\boldsymbol{\Omega}\text{'}_i$ and $\boldsymbol{\Omega}\text{'}_j$), we know that there exists a corresponding $\boldsymbol{q}_{ij} \in \mathbb{R}^{d^2}$ such that we can rewrite Eq. (21) (or Eq. (22)) as:

$$\boldsymbol{q}_{ij}^\top\boldsymbol{\Omega'} - \epsilon \geq 0 \tag{23}$$

where we again defined $\boldsymbol{\Omega'}$ to be the flattened version of $\boldsymbol{\Omega}$'.

Finally, by making use of the Schur complement, we can rewrite Eq. (23) as the following semi-definite constraint:

$$\begin{pmatrix} 1 & 0 \\ 0 & \boldsymbol{q}_{ij}^\top\boldsymbol{\Omega'} - \epsilon \end{pmatrix} \succeq 0 \tag{24}$$

References

1. Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K.T., Montavon, G., Samek, W., Müller, K., Dähne, S., Kindermans, P.: innvestigate neural networks! J. Mach. Learn. Res. **20**, 93:1–93:8 (2019), http://jmlr.org/papers/v20/18-540.html
2. Artelt, A.: Ceml: Counterfactuals for explaining machine learning models - a python toolbox. https://www.github.com/andreArtelt/ceml (2019)
3. Artelt, A., Hammer, B.: Efficient computation of counterfactual explanations of LVQ models. CoRR **abs/1908.00735** (2019), http://arxiv.org/abs/1908.00735
4. Artelt, A., Hammer, B.: On the computation of counterfactual explanations - a survey. ArXiv **abs/1911.07749** (2019)
5. Artelt, A., Hammer, B.: On the computation of counterfactual explanations - A survey. CoRR **abs/1911.07749** (2019), http://arxiv.org/abs/1911.07749
6. Bellet, A., Habrard, A., Sebban, M.: A survey on metric learning for feature vectors and structured data. CoRR **abs/1306.6709** (2013), http://arxiv.org/abs/1306.6709
7. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, New York, NY, USA (2004)
8. Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L.: Explaining explanations: An overview of interpretability of machine learning. In: 5th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2018, Turin, Italy, October 1-3, 2018. pp. 80–89 (2018). https://doi.org/10.1109/DSAA.2018.00018, https://doi.org/10.1109/DSAA.2018.00018
9. Kulis, B.: Metric learning: A survey. Foundations and Trends in Machine Learning **5**(4), 287–364 (2013). https://doi.org/10.1561/2200000019, https://doi.org/10.1561/2200000019
10. Molnar, C.: Interpretable Machine Learning (2019), https://christophm.github.io/interpretable-ml-book/
11. Nova, D., Estévez, P.A.: A review of learning vector quantization classifiers. Neural Comput. Appl. **25**(3-4), 511–524 (Sep 2014). https://doi.org/10.1007/s00521-013-1535-3, https://doi.org/10.1007/s00521-013-1535-3

12. Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., Bie, T.D., Flach, P.A.: FACE: feasible and actionable counterfactual explanations. CoRR **abs/1909.09369** (2019), http://arxiv.org/abs/1909.09369
13. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1135–1144. KDD '16, ACM, New York, NY, USA (2016). https://doi.org/10.1145/2939672.2939778, http://doi.acm.org/10.1145/2939672.2939778
14. Samek, W., Wiegand, T., Müller, K.: Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. CoRR **abs/1708.08296** (2017), http://arxiv.org/abs/1708.08296
15. Schneider, P., Biehl, M., Hammer, B.: Adaptive relevance matrices in learning vector quantization. Neural Computation **21**(12), 3532–3561 (2009). https://doi.org/10.1162/neco.2009.11-08-908, https://doi.org/10.1162/neco.2009.11-08-908, pMID: 19764875
16. Schneider, P., Biehl, M., Hammer, B.: Distance learning in discriminative vector quantization. Neural Computation **21**(10), 2942–2969 (2009). https://doi.org/10.1162/neco.2009.10-08-892, https://doi.org/10.1162/neco.2009.10-08-892, pMID: 19635012
17. Tjoa, E., Guan, C.: A survey on explainable artificial intelligence (XAI): towards medical XAI. CoRR **abs/1907.07374** (2019), http://arxiv.org/abs/1907.07374
18. Wachter, S., Mittelstadt, B.D., Russell, C.: Counterfactual explanations without opening the black box: Automated decisions and the GDPR. CoRR **abs/1711.00399** (2017), http://arxiv.org/abs/1711.00399
19. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. **10**, 207–244 (2009), https://dl.acm.org/citation.cfm?id=1577078