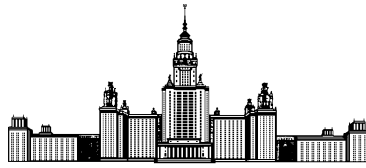


Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

Кафедра Математических Методов Прогнозирования

## Курсовая работа студента 317 группы

### Обзорная статья по теме «Обучение метрики»

Выполнил:

студент 3 курса 317 группы

*Борисов Алексей Антонович*

Научный руководитель:

д.ф.-м.н., профессор

*А. Г. Дьяконов*

Москва, 2022

# Содержание

<b>1</b>	<b>Введение</b>	<b>2</b>
<b>2</b>	<b>Классические методы</b>	<b>4</b>
2.1	Large Margin Nearest Neighbors . . . . .	4
2.2	Nearest Component Analysis . . . . .	5
2.3	Information-Theoretic Metric Learning . . . . .	6
<b>3</b>	<b>Обучение метрики на Римановом многообразии</b>	<b>7</b>
3.1	Обучение метрики в пространстве положительно определенных матриц . . . . .	8
3.2	Обучение метрики на многообразии Грассмана . . . . .	9
3.3	Geometric Mean Metric Learning . . . . .	9
3.4	Обобщенный подход . . . . .	11
<b>4</b>	<b>Применение современных техник</b>	<b>11</b>
4.1	Adversarial Metric Learning . . . . .	11
4.2	Metric Transfer Learning . . . . .	13
<b>5</b>	<b>Некоторые проблемы методов обучения метрики</b>	<b>14</b>
<b>6</b>	<b>Нелинейные подходы при обучении метрики</b>	<b>15</b>
6.1	Ядерное обобщение методов обучения метрики . . . . .	16
6.2	Multi-metric learning . . . . .	17
6.3	Curvilinear metric learning . . . . .	18
<b>7</b>	<b>Заключение</b>	<b>20</b>

# 1 Введение

В общем виде задача обучения метрики (metric learning) заключается в обучении расстояния, использование которого способно улучшить качество работы алгоритмов, основанных на использовании информации о «похожести» объектов (similarity-based algorithms). Примерами таких алгоритмов являются: метод  $k$ -ближайших соседей (K-Nearest Neighbors), метод  $k$ -средних (K-Means).

Пусть  $\mathcal{X} = \{x_1, \dots, x_N\}$  — входной набор данных, на котором определены меры сходства для различных пар и троек данных. Эти меры сходства задаются следующими множествами:

$$S = \{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} : x_i \text{ и } x_j \text{ являются похожими}\},$$

$$D = \{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} : x_i \text{ и } x_j \text{ не являются похожими}\},$$

$$R = \{(x_i, x_j, x_l) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : x_i \text{ является более похожим на } x_j \text{ чем на } x_l\}.$$

Для заданных входных данных  $\mathcal{X}$  с определенными множествами  $S$ ,  $D$  и  $R$  задача заключается в поиске метрики из некоторого семейства  $\mathcal{D}$ , которая является наилучшей в смысле некоторой функции потерь  $\mathcal{L}$ . Таким образом, получаем следующую задачу оптимизации:

$$\min_{d \in \mathcal{D}} \mathcal{L}(d, S, D, R).$$

Если в дополнение к множеству  $\mathcal{X}$  дан набор соответствующих меток  $y_1, \dots, y_N$ , соответствующих элементам из  $\mathcal{X}$ , то множества  $S$ ,  $D$  и  $R$  могут быть представлены в следующем виде:

$$S = \{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} : y_i = y_j\},$$

$$D = \{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} : y_i \neq y_j\},$$

$$R = \{(x_i, x_j, x_l) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : y_i = y_j \neq y_l\}.$$

Также возможен учет различной дополнительной информации о близости объектов, например:

$$S = \{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} : y_i = y_j \text{ и } x_j \in \mathcal{U}(x_i)\}.$$

Здесь  $\mathcal{U}(x_i)$  обозначает множество соседей  $x_i$ , которое может формироваться, исходя из некоторой дополнительной информации.

Множество  $\mathcal{D}$  обычно представляет собой некоторое параметризованное семейство расстояний. Во многих классических методах обучения метрики в качестве такого семейства выступает расстояние Махаланобиса. Если мы работаем в  $\mathbb{R}^d$ , то это семейство параметризуется матрицей  $M \in S_d(\mathbb{R})_0^+$ , где  $S_d(\mathbb{R})_0^+$  обозначает множество неотрицательно определенных

матриц (PSD cone) размера  $d$  и имеет вид:

$$d_M(x, y) = \sqrt{(x - y)^T M (x - y)}, \quad x, y \in \mathbb{R}^d.$$

Тогда в случае  $\mathcal{X} \subset \mathbb{R}^d$ ,  $\mathcal{D} = \{d_M : M \in S_d(\mathbb{R})_0^+\}$  и при наличии меток  $y_1, \dots, y_N$  задача оптимизации принимает следующий вид:

$$\min_{M \in S_d(\mathbb{R})_0^+} \mathcal{L}(d_M, (x_1, y_1), \dots, (x_N, y_N)).$$

Но данную задачу можно параметризовать и по-другому. Из одной из теорем о разложении матриц следует, что если  $M \in S_d(\mathbb{R})_0^+$ , то существует такая матрица  $L \in \mathbb{R}^{d \times d}$ , что  $M = L^T L$ , причем матрица  $L$  единственна с точностью до изометрии. Таким образом:

$$d_M^2(x, y) = (x - y)^T M (x - y) = (x - y)^T L^T L (x - y) = (L(x - y))^T (L(x - y)) = \|L(x - y)\|_2^2,$$

то есть семейство расстояний Махаланобиса можно параметризовать с помощью матрицы  $L \in \mathbb{R}^{d \times d}$  без дополнительных ограничений. В этом случае матрица  $L$  задает линейное отображение  $x \rightarrow Lx$ , такое что евклидово расстояние в этом новом пространстве совпадает с расстоянием Махаланобиса с матрицей  $M = L^T L$  в исходном пространстве.

В случае если матрица  $M$  оказывается неполного ранга, это соответствует обучению расстояния в пространстве меньшей размерности, что позволяет уменьшить размерность входных данных. То же происходит и при обучении линейного отображения  $L$  неполного ранга. Можно расширить этот случай и напрямую обучать отображение  $L \in \mathbb{R}^{d' \times d}$ , где  $d' < d$ . Таким образом мы проецируем исходные данные в пространство размерности не выше  $d'$ .

Оба варианта параметризации имеют свои плюсы и минусы. Параметризация с помощью  $M \in S_d(\mathbb{R})_0^+$  зачастую приводит к выпуклой задаче оптимизации, однако необходимо учитывать условие на матрицу  $M$ . Например, при оптимизации градиентными методами необходимо на каждом шаге осуществлять проекцию матрицы  $M$  на множество неотрицательно определенных матриц. При параметризации с помощью  $L \in \mathbb{R}^{d' \times d}$  далеко не всегда получается выпуклая задача, однако можно напрямую обучать отображения в пространство меньшей размерности.

Стоит уточнить, что данная статья практически не будет касаться методов обучения метрики в глубоком обучении (deep metric learning), хотя многие современные статьи, связанные с обучением метрики, рассказывают о его применении в глубоком обучении.

## 2 Классические методы

Прежде чем переходить к более новым исследованиям в области обучения метрики, расскажем о нескольких классических алгоритмах: Large Margin Nearest Neighbors (LMNN) [2], Nearest Component Analysis (NCA) [3] и Information-Theoretic Metric Learning (ITML) [4].

### 2.1 Large Margin Nearest Neighbors

Алгоритм LMNN направлен на увеличение точности метода  $k$ -ближайших соседей. Для каждого объекта  $x_i$  определяются  $k$  ближайших соседей того же класса что и  $x_i$  (target neighbors). Если  $x_j$  является соседом  $x_i$  по классу (target neighbor), то это обозначается  $j \rightsquigarrow i$ . Соседи не меняются во время всего процесса обучения, и для определения соседей можно использовать дополнительную информацию о данных. Во время обучения LMNN старается поместить каждый объект как можно ближе к его  $k$  ближайшим соседям по классу и при этом не допустить, чтобы объекты других классов попадали в зазор, который определяется соседями. Это позволяет алгоритму локально разделять классы, благодаря чему улучшается качество классификации методом  $k$ -ближайших соседей. Таким образом, функция потерь LMNN состоит из двух частей:

$$\begin{aligned}\varepsilon_{pull}(M) &= \sum_{i=1}^N \sum_{j \rightsquigarrow i} d_M^2(x_i, x_j), \\ \varepsilon_{push}(M) &= \sum_{i=1}^N \sum_{j \rightsquigarrow i} \sum_{l=1}^N (1 - y_{il}) [1 + d_M^2(x_i, x_j) - d_M^2(x_i, x_l)]_+,\end{aligned}$$

где  $y_{il} = 1$  если  $y_i = y_l$ , иначе  $y_{il} = 0$ .  $[z]_+ = \max\{z, 0\}$ .

Объединив части в виде взвешенной суммы, получим функцию потерь следующего вида:

$$\mathcal{L}(M) = (1 - \mu)\varepsilon_{pull}(M) + \mu\varepsilon_{push}(M), \quad \mu \in [0, 1].$$

На рисунке 1 проиллюстрирована основная идея алгоритма LMNN.

Полученная функция потерь является выпуклой, и для ее оптимизации можно использовать субградиентные методы с проекцией на множество неотрицательно определенных матриц. Задачу оптимизации можно аналогично сформулировать относительно линейного отображения  $L$  и оптимизировать с помощью градиентных методов, хотя в этом случае задача получается невыпуклой.

Алгоритм LMNN можно использовать последовательно несколько раз, на каждом этапе получая новую метрику, так как после применения LMNN для каждого объекта можно

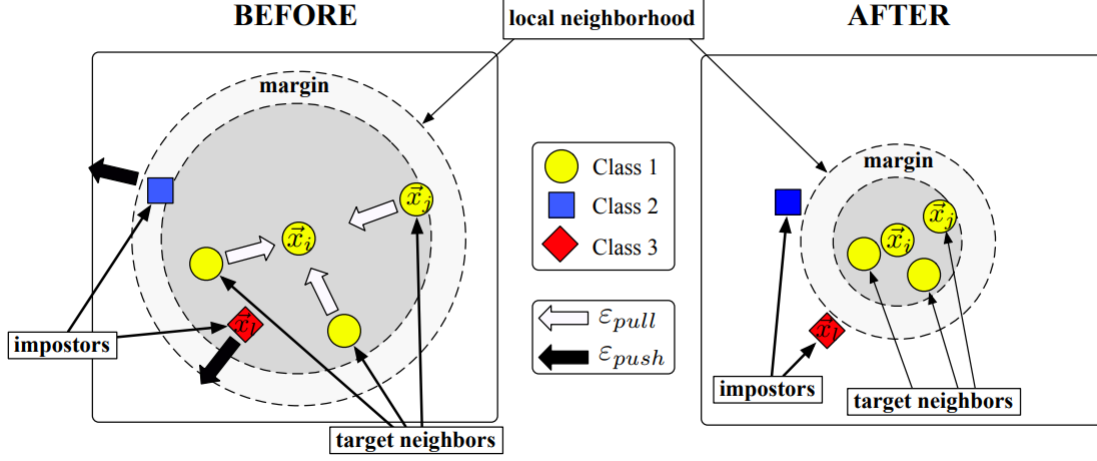


Рис. 1: Основная идея алгоритма LMNN

выбрать более качественное множество соседей по классу.

## 2.2 Nearest Component Analysis

В алгоритме NCA обучается линейное преобразование  $L \in \mathbb{R}^{d \times d}$ , улучшающее точность метода  $k$ -ближайших соседей. Для этого оптимизируется LOO-ошибка (leave-one-out error) для алгоритма ближайшего соседа ( $k = 1$ ). Напрямую оптимизировать LOO-ошибку проблематично, поэтому используется вероятностный подход. Для  $x_i, x_j \in \mathcal{X}$  определяется вероятность того, что  $x_i$  будет иметь  $x_j$  ближайшим соседом в метрике, соответствующей линейному преобразованию  $L$ :

$$p_{ij}^L = \frac{\exp(-\|Lx_i - Lx_j\|^2)}{\sum_{k \neq i} \exp(-\|Lx_i - Lx_k\|^2)}, \quad j \neq i, \quad p_{ii}^L = 0.$$

$p_{i*}$  определяет вероятностную меру на множестве  $\{1, \dots, N\}$ , и естественным образом можно определить вероятность верной классификации  $x_i$ :

$$p_i^L = \sum_{j \in C_i} p_{ij}^L, \quad C_i = \{j \in \{1, \dots, N\} : y_j = y_i\}.$$

Ожидаемое количество верно классифицированных объектов и соответственно функция, которую мы хотим максимизировать, задается следующим образом:

$$f(L) = \sum_{i=1}^N p_i^L = \sum_{i=1}^N \sum_{j \in C_i} p_{ij}^L = \sum_{i=1}^N \sum_{\substack{j \in C_i \\ j \neq i}} \frac{\exp(-\|Lx_i - Lx_j\|^2)}{\sum_{k \neq i} \exp(-\|Lx_i - Lx_k\|^2)}.$$

Данная функция является дифференцируемой и ее можно оптимизировать градиентным спуском. Однако эта задача не является выпуклой, поэтому возможно попадание в локальные

экстремумы. Так как напрямую оптимизируется LOO-ошибка, высока вероятность переобучения.

## 2.3 Information-Theoretic Metric Learning

Алгоритм ITML позволяет учитывать ограничения различного вида, а также априорную информацию о метрике, которую мы хотим обучить. В классическом варианте используется следующий набор ограничений:

$$\begin{aligned} d_M(x_i, x_j) &\leq u, \quad (x_i, x_j) \in S, \\ d_M(x_i, x_j) &\geq l, \quad (x_i, x_j) \in D, \end{aligned}$$

то есть расстояние между похожими объектами не должно превышать некоторого порога  $u$ , а между непохожими объектами расстояние должно быть не меньше порога  $l$ . При этом можно требовать от алгоритма, чтобы обучаемая матрица  $M$  была как можно ближе к заданной матрице  $M_0$ , выбор которой основывается на некоторой информации о природе данных. Например, в случае если распределение данных является нормальным, хорошие результаты может дать использование обратной выборочной ковариационной матрицы в качестве  $M_0$ . При отсутствии дополнительной информации можно использовать единичную матрицу, соответствующую евклидову расстоянию.

Для оценки близости матриц  $M_0$  и  $M$  используется следующий факт: существует биекция между множеством расстояний Махаланобиса и множеством многомерных нормальных распределений с одинаковым средним (не ограничивая общности будем считать, что все рассматриваемые нормальные распределения имеют среднее  $\mu$ ). Так, расстоянию Махаланобиса с матрицей  $M$  соответствует многомерное нормальное распределение:

$$p(\mathbf{x}; M) = \frac{1}{C} \exp(-\frac{1}{2} d_M(\mathbf{x}, \mu)), \quad \text{где } C \text{ это нормировочная константа.}$$

Близость между матрицами  $M_0$  и  $M$  вычисляется, используя расстояние Кульбака-Лейблера между соответствующими многомерными нормальными распределениями:

$$KL(p(\mathbf{x}; M_0) \| p(\mathbf{x}; M)) = \int p(\mathbf{x}; M_0) \log \frac{p(\mathbf{x}; M_0)}{p(\mathbf{x}; M)} d\mathbf{x}$$

Таким образом, оптимизационная задача для метода ITML принимает вид:

$$\min_{M \succeq 0} KL(p(\mathbf{x}; M_0) \| p(\mathbf{x}; M)),$$

при условии  $d_M(x_i, x_j) \leq u, \quad (x_i, x_j) \in S$  и  $d_M(x_i, x_j) \geq l, \quad (x_i, x_j) \in D$ .

Вообще говоря, можно использовать другие условия, например, учитывающие информацию об относительной похожести между объектами (тройки из множества  $R$ ). Для расстояния Кульбака-Лейблера между многомерными нормальными распределениями верно следующее равенство:

$$KL(p(\mathbf{x}; M_0) \| p(\mathbf{x}; M)) = \frac{1}{2} D_{ld}(M_0^{-1}, M^{-1}) = \frac{1}{2} D_{ld}(M, M_0),$$

где  $D_{ld}(M, M_0) = \text{tr}(MM_0^{-1}) - \log \det(MM_0^{-1}) - n$ , LogDet дивергенция.

Так, задача сводится к следующей оптимизации LogDet дивергенции:

$$\min_{M \succeq 0} D_{ld}(M, M_0),$$

$$\text{tr}(M(x_i - x_j)(x_i - x_j)^T) \leq u, \quad (x_i, x_j) \in S,$$

$$\text{tr}(M(x_i - x_j)(x_i - x_j)^T) \geq l, \quad (x_i, x_j) \in D.$$

В оригинальной статье авторы предоставляют итерационный алгоритм, решающий полученную оптимизационную задачу.

Было придумано еще большое количество различных алгоритмов, в которых из некоторых эмпирических соображений выводилась новая задача оптимизации. Например, Average Neighborhood Margin Maximization (ANMM) [5], Nearest Class Mean Metric Learning (NCMML) [6], Maximally Collapsing Metric Learning (MCML) [7]. Также многие алгоритмы, например, LMNN, ITML, ANMM имеют ядерные обобщения, что позволяет искать оптимальную метрику в другом пространстве, обычно большей размерности.

### 3 Обучение метрики на Римановом многообразии

Можно выделить целое направление работ по обучению метрики, в которых поиск расстояния осуществляется не в привычном евклидовом пространстве, а в пространстве более сложной структуры. Обычно в качестве такого пространства выступает пространство положительно определенных матриц (SPD manifold) или многообразие Грассмана (Grassman manifold). Они оба являются частными случаями многообразия Римана.



### 3.1 Обучение метрики в пространстве положительно определенных матриц

Пусть  $\{X_1, \dots, X_n\}, X_i \in \mathbb{R}^{d \times m_i}$ , где  $d$  — размерность пространства признаков, а  $m_i$  — количество объектов.  $X_i$  соответствует значению целевой переменной  $l_i$ . В качестве  $X_i$  может выступать набор фотографий или множество локальных признаков, извлеченных из изображения. По  $\{X_1, \dots, X_n\}$  может быть сформирован набор положительно определенных матриц  $\{S_1, \dots, S_n\}, S_i \in \mathbb{R}^{d \times d}$ , где  $S_i$  некоторым образом агрегирует информацию об объектах из  $X_i$ . Например, в статье про Log-Euclidean Metric Learning (LEML) [8] матрица  $S_i$  имеет размер  $d + 1$  и формируется следующим образом:

$$S_i = |\mathbf{S}|^{-\frac{1}{d+1}} \begin{bmatrix} \mathbf{S} + \mathbf{m}\mathbf{m}^T & \mathbf{m} \\ \mathbf{m}^T & 1 \end{bmatrix},$$

где  $\mathbf{m}$  — среднее, а  $\mathbf{S}$  — матрица ковариации соответствующие  $X_i$ .

Задача заключается в том, чтобы найти расстояние на множестве положительно определенных матриц, удовлетворяющее некоторым полезным свойствам, например,  $d(S_i, S_j) \leq u$ , если  $S_i$  и  $S_j$  соответствуют одной и той же целевой переменной, иначе  $d(S_i, S_j) \geq l$ . В более ранних статьях рассматривался вариант отображения  $d \times d$  положительно определенных матриц в пространство  $\mathbb{R}^{\frac{d(d+1)}{2}}$  с последующим обучением расстояния Махаланобиса. Но этот вариант плохо учитывает геометрию исходного пространства. В уже упомянутом методе LEML обучается матрица, осуществляющая отображение в пространство положительно определенных матриц меньшей размерности (или совпадающей с исходной), и уже в нем вычисляется расстояние, порожденное нормой Фробениуса.

Было создано большое количество подходов для обучения метрики в пространстве положительно определенных матриц, например, Affine Invariant Riemannian Metric (AIRM) [9], Log-Euclidean Riemannian Metric (LERM) [10]. Возможно использование ядерных методов для отображения пространства положительно определенных матриц в гильбертово спрямляющее пространство (Reproducing kernel Hilbert space), в котором уже используются методы для разделения объектов разных классов [11, 12].

### 3.2 Обучение метрики на многообразии Грассмана

При работе в многообразии Грассмана вместо положительно определенных матриц строятся  $q$ -мерные линейные подпространства, натянутые на ортонормированный базис, задающийся в виде матрицы  $Y_i \in \mathbb{R}^{d \times q}$ . Эти подпространства строятся следующим образом:

$$X_i X_i^T \simeq Y_i \Lambda_i Y_i^T,$$

где  $\Lambda_i$  соответствует матрица с  $q$  наибольшими собственными значениями, а  $Y_i$  матрица с соответствующими собственными векторами.

Обычно производится отображение из многообразия Грассмана в некоторое гильбертово пространство, в котором обучается проекция в пространство  $\mathbb{R}^d$ , в котором удается разделить объекты разных классов [13, 14]. Однако, в статье [15] обучается отображение в другое многообразие Грассмана, в котором объекты разных классов лучше разделяются. Иллюстрация обоих подходов изображена на рисунке 2.

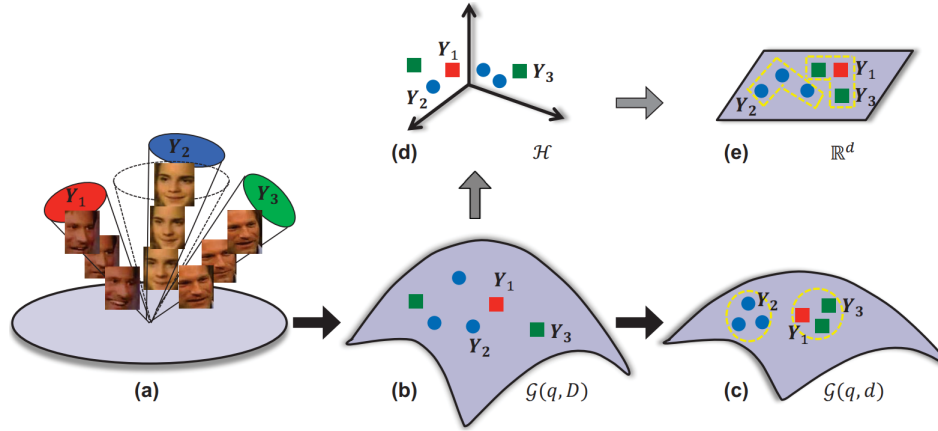


Рис. 2: Иллюстрация основной идеи методов обучения метрики, использующих многообразие Грассмана. (a)-(b)-(d)-(e) представляет собой первый вариант, с отображением в гильбертово пространство  $\mathcal{H}$ , а (a)-(b)-(c) вариант с отображением в другое многообразие Грассмана.

### 3.3 Geometric Mean Metric Learning

Очень интересной является следующая статья [16]. В ней авторы используют новую функцию потерь, которая сильно отличается от тех, что использовались прежде. Обычно функция потерь, которую используют для обучения расстояния Махаланобиса с матрицей  $\mathbf{M}$  преследует следующие две идеи: расстояние между похожими объектами в метрике, порожденной матрицей  $\mathbf{M}$ , нужно минимизировать, а расстояние между непохожими объектами нужно

максимизировать, причем, разумеется, в той же метрике. Однако в данной статье минимизируют следующую функцию потерь:

$$\mathcal{L}(\mathbf{M}) = \sum_{(x_i, x_j) \in S} d_{\mathbf{M}}(x_i, x_j) + \sum_{(x_i, x_j) \in D} d_{\mathbf{M}^{-1}}(x_i, x_j),$$

где  $S$  и  $D$  представляют собой множества пар похожих и непохожих объектов соответственно.

За этой функцией потерь стоит следующая интуиция: градиенты  $\frac{\partial d_{\mathbf{M}}}{\partial \mathbf{M}}$  и  $\frac{\partial d_{\mathbf{M}^{-1}}}{\partial \mathbf{M}}$  направлены почти в противоположном направлении, поэтому расстояние между непохожими объектами  $d_{\mathbf{M}}(x_i, x_j)$  может увеличиться при уменьшении  $d_{\mathbf{M}^{-1}}(x_i, x_j)$ .

После введения следующих обозначений:

$$\begin{aligned} \mathbf{S} &= \sum_{(x_i, x_j) \in S} (x_i - x_j)(x_i - x_j)^T, \\ \mathbf{D} &= \sum_{(x_i, x_j) \in D} (x_i - x_j)(x_i - x_j)^T, \end{aligned}$$

оптимизационную задачу можно записать в следующем виде:

$$\min_{\mathbf{M} \succeq 0} \mathcal{L}(\mathbf{M}) = \text{tr}(\mathbf{M}\mathbf{S}) + \text{tr}(\mathbf{M}^{-1}\mathbf{D}).$$

Вместо того чтобы оптимизировать данную функцию потерь итерационными методами, в статье доказывается ее строгая выпуклость и строгая геодезическая выпуклость в пространстве положительно определенных матриц. После нахождения градиента функции потерь и приравнивания его нулю задача сводится к решению следующего уравнения Риккати:  $\mathbf{M}\mathbf{S}\mathbf{M} = \mathbf{D}$ . Единственным решением этого уравнения является середина геодезической кривой, соединяющей  $\mathbf{S}^{-1}$  и  $\mathbf{D}$ , которая определяется следующим образом:  $\mathbf{A}^\sharp_t \mathbf{B} = \mathbf{A}^{1/2}(\mathbf{A}^{-1/2} \mathbf{B} \mathbf{A}^{-1/2})^t \mathbf{A}^{1/2}$ ,  $t \in [0, 1]$ .

На рисунке 3 изображено решение поставленной задачи, им является матрица, расположенная на середине геодезической кривой, соединяющей матрицы  $\mathbf{S}^{-1}$  и  $\mathbf{D}$  в пространстве положительно определенных матриц. Таким образом, решение поставленной оптимизационной задачи можно записать в явном виде:

$$\mathbf{M} = \mathbf{S}^{-1} \sharp_{1/2} \mathbf{D} = \mathbf{S}^{-1/2} (\mathbf{S}^{1/2} \mathbf{D} \mathbf{S}^{1/2})^{1/2} \mathbf{S}^{-1/2}.$$

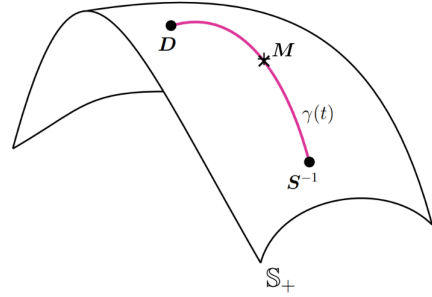


Рис. 3: Решение задачи GMML расположено в середине геодезической кривой, соединяющей матрицы  $\mathbf{S}^{-1}$  и  $\mathbf{D}$  в пространстве положительно определенных матриц.

### 3.4 Обобщенный подход

Как можно заметить, методы обучения метрики в пространстве положительно определенных матриц и на многообразии Грассмана имеют много общего. Так в статье [17], вдохновившись подходом, использованным в Geometric Mean Metric Learning (GMML), авторы показали единый метод для поиска расстояния в вышеуказанных пространствах.

Вводится ряд обозначений. Для пространства положительно определенных матриц:

$$\mathbf{S} = \sum_{(x_i, x_j) \in S} (\mathbf{T}_i - \mathbf{T}_j)(\mathbf{T}_i - \mathbf{T}_j), \quad \mathbf{D} = \sum_{(x_i, x_j) \in d} (\mathbf{T}_i - \mathbf{T}_j)(\mathbf{T}_i - \mathbf{T}_j),$$

где  $\mathbf{T}_i = \log S_i$  или  $S_i^{1/2}$ .

А для многообразия Грассмана:

$$\mathbf{S} = \sum_{(x_i, x_j) \in S} \mathbf{T}_{ij} \mathbf{T}_{ij}, \quad \mathbf{D} = \sum_{(x_i, x_j) \in D} \mathbf{T}_{ij} \mathbf{T}_{ij}, \quad \text{где } \mathbf{T}_{ij} = Y_i Y_i^T - Y_j Y_j^T.$$

Тогда, используя функцию потерь из метода GMML, в обоих случаях оптимизационную задачу можно записать в следующем виде:

$$\min_{\mathbf{M} \succeq 0} \mathcal{L}(\mathbf{M}) = \text{tr}(\mathbf{M} \mathbf{S}) + \text{tr}(\mathbf{M}^{-1} \mathbf{D}).$$

Решение данной задачи можно записать в явном виде:  $\mathbf{M} = \mathbf{S}^{-1} \sharp_{1/2} \mathbf{D}$ . Однако вычисление геометрического среднего является довольно трудоемким, поэтому авторы предлагают использовать упомянутый ранее эффективный LERM подход. Для обеспечения положительной определенности матриц  $\mathbf{S}$  и  $\mathbf{D}$  добавляется регуляризация с параметром  $\lambda \geq 0$ , а также вводится параметр  $t$ , позволяющий увеличивать влияние пар похожих или непохожих объектов на итоговый ответ. Таким образом, итоговое решение записывается в следующем виде:

$$\mathbf{M}_{final} = \exp \left( \frac{-t \log(\mathbf{S} + \lambda \mathbf{I}) + (1 - t) \log(\mathbf{D} + \lambda \mathbf{I})}{2} \right).$$

## 4 Применение современных техник

В этом разделе речь пойдет об использовании при обучении метрики подходов, хорошо зарекомендовавших себя в других областях машинного обучения.

### 4.1 Adversarial Metric Learning

В статье [18] для обеспечения лучшей робастности и дискриминативной способности алгоритмов обучения метрики предлагается использовать состязательное обучение (adversarial

training).

Введем некоторые обозначения,  $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_N] \in \mathbb{R}^{2d \times N}$  — матрица, содержащая  $N$  обучающих пар, где  $\mathbf{X}_i = [\mathbf{x}_i^T, \mathbf{x}_i'^T] \in \mathbb{R}^{2d}$ . Матрице пар соответствует вектор меток  $\mathbf{y} \in \{-1, 1\}^N$ , в котором  $y_i = 1$ , если  $\mathbf{x}_i^T$  и  $\mathbf{x}_i'^T$  похожи и  $y_i = -1$  иначе.

В отличие от классических методов обучения метрики алгоритм AML состоит из двух этапов: *confusion* и *distinguishment*.

На этапе *confusion* алгоритм старается сгенерировать такие пары  $\Pi$ , чтобы ввести обучаемую метрику  $\mathbf{M}$  в заблуждение и заставить ее ошибаться на этих парах. При этом пара  $\Pi_i$  должна быть как можно ближе к исходной паре  $\mathbf{X}_i$ . Данные требования записываются в виде следующей оптимизационной задачи:

$$\min_{\Pi} C_{\mathbf{M}}(\Pi) = \mathcal{L}(\mathbf{M}, \Pi, -\mathbf{y}) - \beta \text{Dist}_{\mathbf{M}}^+(\mathbf{X}, \Pi),$$

где  $\text{Dist}_{\mathbf{M}}^+(\mathbf{X}, \Pi) = \sum_{i=1}^N d_{\mathbf{M}}(\mathbf{x}_i, \pi_i) + \sum_{i=1}^N d_{\mathbf{M}}(\mathbf{x}_i', \pi_i')$ .

На рисунке 4 показано, каким образом генерируются новые пары.

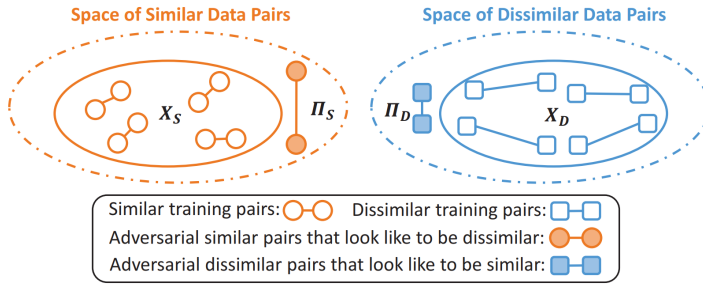


Рис. 4: Процесс генерации новых пар в методе AML.

В свою очередь на этапе *distinguishment* алгоритм должен обновить метрику  $\mathbf{M}$  так, чтобы минимизировать число пар, на которых он ошибается, причем пары берутся как из исходной обучающей выборки, так и сгенерированные на этапе *confusion*. Это записывается в виде следующей оптимизационной задачи:

$$\min_{\mathbf{M} \succeq 0} D_{\Pi}(\mathbf{M}) = \mathcal{L}(\mathbf{M}, \mathbf{X}, \mathbf{y}) + \alpha \mathcal{L}(\mathbf{M}, \Pi, \mathbf{y}).$$

Коэффициенты  $\alpha$  и  $\beta$  определяют влияние соответствующих слагаемых при оптимизации.

Таким образом, получаем алгоритм, в котором поочередно выполняются этапы *confusion* и *distinguishment*:

$$\begin{cases} \mathbf{M}^{(t+1)} = \arg \min D_{\Pi^{(t)}}(\mathbf{M}^{(t)}), (Distinguishment), \\ \Pi^{(t+1)} = \arg \min C_{\mathbf{M}^{(t+1)}}(\Pi^{(t)}), (Confusion). \end{cases}$$

Однако для получившейся задачи не гарантируется сходимость, и авторы статьи свели ее

похожей задаче, для которой сходимость гарантируется:

$$\min_{M \succeq 0, \Pi^*} D_{\Pi^*}(M), \quad \text{при условии } \Pi^* = \arg \min C_M(\Pi).$$

При этом функция  $C_M(\Pi)$  должна быть строго квазивыпуклой, что гарантирует единственность  $\Pi^*$ . Для этого авторы использовали функцию потерь, которая использовалась в GMM-L. Для нее получившуюся задачу можно решать градиентным спуском с проекциями, для которого гарантируется сходимость.

## 4.2 Metric Transfer Learning

Подавляющее большинство методов машинного обучения предполагает, что объекты обучающей и тестовой выборки принадлежат одному распределению. Однако во многих ситуациях это предположение может быть не выполнено, например, имеется большое количество размеченных данных из распределения, которое отличается от целевого (к которому мы хотим применять модель). Тогда, для того чтобы побороться с различием в распределениях в исходном и целевом доменах могут быть использованы методы transfer learning. Также может существовать необходимость найти подходящую меру близости между объектами из целевого домена, здесь возникает задача metric transfer learning.

Интересующие нас методы transfer learning можно разделить на несколько категорий: instance-based, feature-based и metric-based.

В instance-based подходах осуществляется перевзвешивание объектов из исходного домена, для того чтобы после перевзвешивания их можно было использовать для обучения модели, применяемой для объектов из целевого домена.

В feature-based подходах осуществляется поиск такого преобразования признаков, которое будет минимизировать расстояние между распределениями объектов из исходного и целевого доменов. Основная проблема данного подхода для задачи поиска меры близости заключается в том, что при применении преобразования признаков теряется внутренняя геометрическая структура данных.

Metric-based подходы, как следует из названия, ориентированы именно на поиск метрики для объектов из целевого домена. Например, в алгоритме transfer metric learning (TML) [19] формулируется задача схожая с задачей multi-task metric learning, и метрика для целевого домена обучается, используя метрики для других задач и корреляции между целевой задачей

и остальными. В еще одном подходе, Deep Transfer Metric Learning (DTML) [20], предлагается использовать нейронные сети для обучения набора иерархических нелинейных преобразований для переноса дискриминативной способности с размеченных данных из исходного домена на неразмеченный целевой домен. Для оптимизации в этом методе предлагается использовать критерий Maximum Mean Discrepancy (MMD). На рисунке 5 отражена основная идея алгоритма DTML.

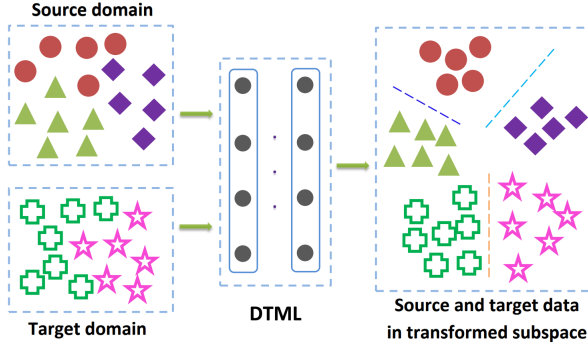


Рис. 5: Основная идея алгоритма DTML.

В более современном подходе, который авторы назвали Metric Transfer Learning Framework (MTLF) [21], представлен подход, который заключается в одновременном обучении как весов для объектов обучающей выборки, так и метрики Махаланобиса. Эта особенность является одним из основных отличий этого метода от более старого мето-

да consistent distance metric learning (CDML) [22], в котором сначала обучались веса объектов под использование с евклидовой метрикой, а затем на перевзвешенных объектах обучалась метрика.

## 5 Некоторые проблемы методов обучения метрики

Как было показано выше, многие методы обучения метрики зачастую используют метрику Махаланобиса и обучают либо соответствующую положительно определенную матрицу  $M$ , либо порожденное ей линейное преобразование  $L$ , где  $L^T L = M$ . Но с ростом числа объектов в обучающей выборке  $N$  или большой размерности данных  $D$  использование этих методов сопряжено с некоторыми проблемами.

Существенной проблемой является положительная определенность матрицы  $M$ . Методы, использующие разновидности градиентного спуска с проекциями на множество положительно определенных матриц (PSD cone) быстро становятся вычислительно неэффективными из-за того, что для осуществления проекции приходится находить собственные значения и вектора матрицы  $M$ .

Методы обучения метрики зачастую обучаются на парах или тройках объектов, поэтому с ростом  $N$  количество обучающих примеров очень быстро растет. Для решения этой проблемы используют онлайн-методы обучения, например, стохастический градиентный спуск (SGD). Однако как было отмечено выше, это не решает проблемы с необходимостью осуществлять проекцию матрицы на PSD cone.

Количество элементов матрицы, которые являются параметрами алгоритма, квадратично зависит от числа признаков, поэтому для уменьшения вероятности переобучения и добавления некоторой регуляризации может быть необходимо вводить ограничение на ранг обучаемой матрицы, что также имеет свои трудности при формулировке задачи оптимизации относительно матрицы  $M$ .

Существует большое количество подходов для уменьшения вычислительной сложности, связанной с ограничением на положительную определенность матрицы  $M$ , например, можно осуществлять проекцию не на каждой итерации, а только раз в несколько итераций. В методе DML-eig [23] описывается алгоритм обучения, для которого на каждой итерации необходимо лишь минимальное собственное значение и соответствующий собственный вектор. Для их вычисления предлагается использовать более эффективный алгоритм Франка-Вульфа. А в статье [24] шаг проекции заменяется решением задачи линейного программирования с ограничениями. Однако вычислительная сложность одной итерации даже с использованием описанных техник остается достаточно высокой. В статье [25] описывается метод Large-Margin Distance Metric Learning (LMDML), имеющий схожие черты с LMNN. Но основным достоинством данной статьи является используемый алгоритм обучения, позволяющий не осуществлять проекцию матрицы  $M$  на множество положительно определенных матриц. Этот эффект достигается за счет того, что на каждой итерации вычисляется размер допустимого шага, после осуществления которого матрица  $M$  останется положительно определенной.

## 6 Нелинейные подходы при обучении метрики

В большинстве подходов обучения метрики в качестве параметризованного множества расстояний используется расстояние Махаланобиса, но оно обладает существенным недостатком. Как уже было сказано, использование расстояния Махаланобиса равносильно использованию евклидова расстояния в пространстве, полученном из исходного с помощью



линейного отображения. Таким образом, методы, обучающие одну метрику Махаланобиса, не способны учесть нелинейные зависимости в данных. В данном разделе представлены несколько подходов обучения метрики, которые стремятся побороть это существенное ограничение.

## 6.1 Ядерное обобщение методов обучения метрики

Ядерное обобщение в методах обучения метрики осуществляется так же как в методе опорных векторов (SVM). Основная идея заключается в отображении исходных данных в пространство большей (возможно бесконечной) размерности, используя отображение  $\phi : \mathbb{R}^d \rightarrow \mathcal{F}$ , где  $\mathcal{F}$  — Гильбертово спрямляющее пространство (Reproducing kernel Hilbert space). Для обучения метрики, обучается линейный непрерывный оператор  $L : \mathcal{F} \rightarrow \mathbb{R}^{d'}$ , где  $d' < d$  и  $L \in \mathcal{L}(\mathcal{F}, \mathbb{R}^{d'})$ , где  $\mathcal{L}(\mathcal{F}, \mathbb{R}^{d'})$  — множество всех линейных непрерывных операторов из  $\mathcal{F}$  в  $\mathbb{R}^{d'}$ . При отображении данных в спрямляющее пространство возникают проблемы, связанные с большой размерностью данных, из-за чего применение алгоритмов может стать вычислительно трудоемким, а в случае бесконечной размерности спрямляющего пространства даже невозможным. Как и в случае метода опорных векторов используется *kernel trick*: определяется функция ядра  $K : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ , такая что  $K(x, x') = \langle \phi(x), \phi(x') \rangle$ . Использование *kernel trick* в метрических алгоритмах возможно благодаря тому, что евклидово расстояние в спрямляющем пространстве можно записать используя только функцию ядра:

$$\begin{aligned} \|\phi(x) - \phi(x')\|^2 &= \langle \phi(x) - \phi(x'), \phi(x) - \phi(x') \rangle \\ &= \langle \phi(x), \phi(x) \rangle - 2\langle \phi(x), \phi(x') \rangle + \langle \phi(x'), \phi(x') \rangle \\ &= K(x, x) + K(x', x') - 2K(x, x'). \end{aligned}$$

Появляются проблемы с обучением отображения  $L \in \mathcal{L}(\mathcal{F}, \mathbb{R}^{d'})$ , так как оно не обязательно представимо в виде матрицы, а даже если представимо, то ее размерность может быть слишком большой. Так как отображение  $L$  является линейным непрерывным функционалом, то для него верна теорема Риса-Фреше о представлении линейного непрерывного функционала в виде скалярного произведения. Но для большинства алгоритмов обучения метрики, для которых применяется ядерное обобщение, доказываются специальные теоремы представления, позволяющие записать отображение  $L$  в следующем виде:

$$L\phi(x) = A \begin{pmatrix} K(x_1, x) \\ \dots \\ K(x_N, x) \end{pmatrix}, \quad \text{где } A \in \mathbb{R}^{d' \times N}.$$

Благодаря данному представлению, обучение отображения  $L$  заключается в поиске коэффициентов матрицы  $A$ . Для применения преобразования  $L$  к новому объекту  $x$  необходимо вычислить значение функции ядра от этого объекта и каждого объекта из обучающего множества. Перечислим несколько методов обучения метрики, допускающие ядерное обобщение: Large Margin Nearest Neighbors (LMNN), Average Neighborhood Margin Maximization (ANMM), Distance Metric Learning through the Maximization of the Jeffrey Divergence (DMLMJ), Linear Discriminant Analysis (LDA).

## 6.2 Multi-metric learning

Еще один подход, позволяющий учитывать нелинейные зависимости в данных, заключается в обучении множества локальных метрик, которые соответствуют локальным линейным отображениям. В статье посвященной LMNN предоставляется обобщение классического алгоритма с использованием нескольких метрик.

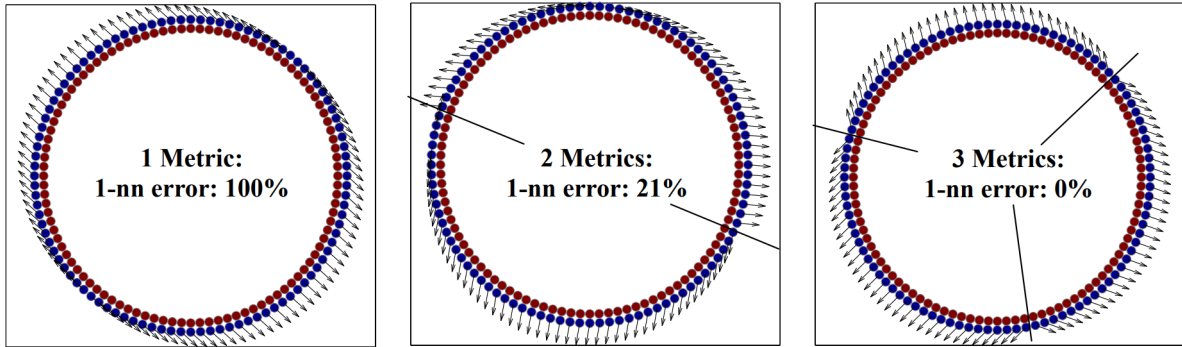


Рис. 6: Искусственный пример, показывающий целесообразность использования нескольких локальных метрик. Стрелками показаны основные направления изменения взаимного расположения объектов при применении линейного отображения соответствующего расстоянию Махаланобиса.

Данное обобщение заключается в том, что сперва проводится кластеризация данных, используя, например, алгоритм  $k$ -средних, спектральную кластеризацию или информацию о классах объектов. Далее для каждого кластера обучается метрика Махаланобиса. На рисунке 6 показан пример, в котором из-за нелинейной границы между классами использование

классического алгоритма LMNN не позволяет достичь хороших результатов, но при кластеризации алгоритмом  $k$ -средних и последующем обучении трех локальных метрик удастся получить более качественную метрику.

Пусть  $c$  — число кластеров, совпадающее с числом классов,  $M^1, \dots, M^c$  — обучаемые метрики, тогда объекту  $x_i$  класса  $y_i$  соответствует метрика с матрицей  $M^{y_i}$ . Расстояние от объекта  $x_i$  до объекта  $x_j$ :

$$d(x_i, x_j) = (x_i - x_j)^T M^{y_j} (x_i - x_j),$$

причем стоит заметить, что данная функция не является симметричной.

В оригинальной статье [2], посвященной LMNN, формулируется задача полуопределенного программирования (semidefinite programming) для multi-metric learning, представляющая собой модифицированную версию задачи для классического метода LMNN.

В статье [26] представлен единый подход к обучению нескольких метрик для более качественного описания «схожести» объектов. Авторы назвали свой метод Unified Multi-Metric Learning (UM<sup>2</sup>L). В данном методе обучается набор из  $K$  метрик Махаланобиса  $\mathcal{M}_K = \{M_1, M_2, \dots, M_K\}$ . Между каждой парой объектов можно посчитать их похожесть относительно метрики  $M_k$ :  $f_{M_k}(x_i, x_j) = -d_{M_k}^2(x_i, x_j)$ . Таким образом, мы получаем набор коэффициентов схожести  $f_{\mathcal{M}_K} = \{f_{M_k}\}_{k=1}^K$ . Далее предлагается некоторым образом их агрегировать для похожих и непохожих пар объектов, используя функции  $\kappa_1(\cdot)$  и  $\kappa_2(\cdot)$ , в статье описаны некоторые варианты, которые могут быть наиболее полезны на практике. Авторы утверждают, что подобный подход, в отличие от вышеописанного подхода с кластеризацией и последующим обучением локальных метрик, способен учитывать не только пространственные связи, но и семантические (semantic linkages).

### 6.3 Curvilinear metric learning

В статье [27] представлен метод, позволяющий учитывать нелинейные зависимости в данных. Важную часть при описании этого метода занимает геометрическая интерпретация обучения метрики. Положительно определенную матрицу  $\mathbf{M} \in \mathbb{R}^{d \times d}$ , задающую метрику Махаланобиса, можно представить в следующем виде:  $\mathbf{M} = \mathbf{P}\mathbf{P}^T$ , где  $\mathbf{P} \in \mathbb{R}^{d \times m}$ . Тогда для квадрата расстояния между точками  $\mathbf{x}$  и  $\hat{\mathbf{x}}$  верна следующая запись:

$$d_{\mathbf{M}}^2(\mathbf{x}, \hat{\mathbf{x}}) = (\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{M} (\mathbf{x} - \hat{\mathbf{x}}) = (\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{P}\mathbf{P}^T (\mathbf{x} - \hat{\mathbf{x}}).$$

Обозначим  $\mathbf{p}_i \in \mathbb{R}^d$   $i$ -ый столбец матрицы  $\mathbf{P}$ , тогда  $d_M^2(\mathbf{x}, \hat{\mathbf{x}})$  можно записать в следующем виде:

$$d_M^2(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^m \|\mathbf{p}_i\|_2^2 \|\mathbf{p}_i T_i(\mathbf{x}) - \mathbf{p}_i T_i(\hat{\mathbf{x}})\|_2^2 = \sum_{i=1}^m \|\mathbf{p}_i\|_2^2 \left( \int_{T_i(\mathbf{x})}^{T_i(\hat{\mathbf{x}})} \|\mathbf{p}_i\|_2 dt \right)^2,$$

где  $T_i(\mathbf{x})$  и  $T_i(\hat{\mathbf{x}})$  — проекции точек  $\mathbf{x}$  и  $\hat{\mathbf{x}}$  на прямые, которые задаются векторами  $\mathbf{p}_i$ .  $T_i(\mathbf{x})$  можно записать в следующем виде:  $T_i(\mathbf{x}) = \arg \min_{t \in \mathbb{R}} \|\mathbf{p}_i t - \mathbf{x}\|_2^2$ .

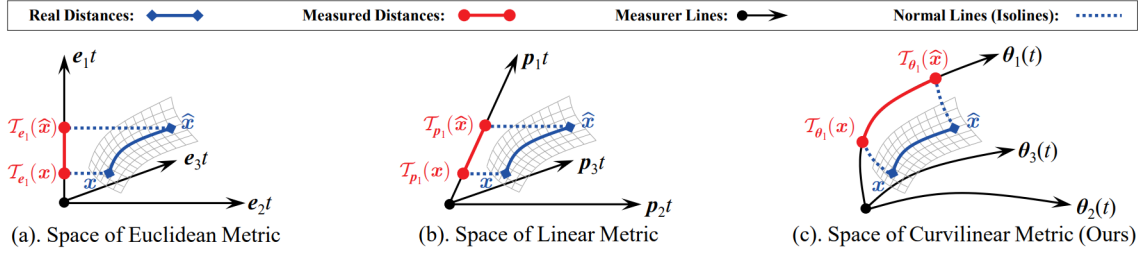


Рис. 7: Иллюстрация евклидовой метрики, метрики Махаланобиса (линейной метрики) и криволинейной метрики в трехмерном пространстве.

Таким образом, расстояние Махаланобиса между точками  $\mathbf{x}$  и  $\hat{\mathbf{x}}$  равно суммарной длине дуг, соединяющих проекции этих точек на прямые  $\mathbf{z}_i(t) = \mathbf{p}_i t$ . Эта интерпретация расстояния Махаланобиса изображена на рисунке 7.

Авторы статьи предлагают более общий подход, в котором вместо прямых  $\mathbf{z}_i(t) = \mathbf{p}_i t$  используются вектор-функции  $\boldsymbol{\theta}_i : \mathbb{R} \rightarrow \mathbb{R}^d$ , которые должны проходить через начало координат при  $t = 0$ , то есть  $\boldsymbol{\theta}_i(0) = \mathbf{0}$ . Аналогично случаю прямых определяется проекция  $\mathcal{T}_i(\mathbf{x})$  точки  $\mathbf{x}$  на кривую  $\boldsymbol{\theta}_i(t)$ . Напомним, что длина дуги между точками  $T_1$  и  $T_2$  на кривой  $\boldsymbol{\theta}_i$  определяется следующим образом:

$$\text{Length}_{\boldsymbol{\theta}_i}(T_1, T_2) = \int_{\min(T_1, T_2)}^{\max(T_1, T_2)} \|\boldsymbol{\theta}_i'(t)\|_2 dt,$$

где  $\boldsymbol{\theta}_i'(t)$  — вектор производных.

Для набора кривых  $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_m)$  расстояние между точками  $\mathbf{x}$  и  $\hat{\mathbf{x}}$  вычисляется следующим образом:

$$d_{\boldsymbol{\Theta}}^2(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^m s_{\boldsymbol{\theta}_i} \cdot \text{Length}_{\boldsymbol{\theta}_i}(\mathcal{T}_{\boldsymbol{\theta}_i}(\mathbf{x}), \mathcal{T}_{\boldsymbol{\theta}_i}(\hat{\mathbf{x}}))^2,$$

где  $s_{\boldsymbol{\theta}_i} = \text{Length}_{\boldsymbol{\theta}_i}^2(0, 1)$ .

Задача обучения криволинейной метрики (Curvilinear Distance Metric Learning, CDML) в общем виде может быть записана следующим образом:

$$\min_{\Theta \in \mathbb{F}_m} \frac{1}{N} \mathcal{L}(d_{\Theta}^2(\mathbf{x}_j, \hat{\mathbf{x}}_j); y_j) + \lambda \mathcal{R}(\Theta),$$

где  $\mathbb{F}_m = \{(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_m) | \theta_{ik}(0) = 0 \text{ и } \theta_{ik}(t) \text{ гладкие}\}$ .

Для того чтобы искать оптимальное значение параметра  $\Theta$ , его необходимо некоторым образом параметризовать, например, функции  $\theta_{ik}$  могут представлять собой полиномы заданной степени или разложения Фурье. Используя теорему Вейерштрасса о возможности приближения произвольной непрерывной функции полиномом, в статье в качестве обучаемого параметра используют трехмерный тензор  $\mathcal{M}$ , который содержит коэффициенты многочленов приближающих функции  $\theta_{ik}(t)$ . Для оптимизации получившейся задачи авторы предлагают использовать стохастический градиентный спуск (SGD), в котором на каждой итерации необходимо находить корни нескольких полиномов для нахождения проекций точек на соответствующие кривые. Также в статье доказывается, что построенная функция  $d_{\Theta}^2(\mathbf{x}, \hat{\mathbf{x}})$  является псевдометрикой, а при дополнительных условиях и полноценной метрикой.

По [ссылке](#) представлена, реализация данного метода, выполненная автором этой обзорной статьи.

## 7 Заключение

В статье описана задача обучения метрики, представлены примеры классических методов решения поставленной задачи. Были описаны некоторые проблемы, которые возникают при решении задачи обучения метрики, а также рассмотрены некоторые более современные подходы к решению задачи. Отдельное внимание было уделено нелинейным методам, которые позволяют учитывать более сложные зависимости в данных.

Современное развитие методов обучения метрики связано прежде всего с развитием глубоких нейронных сетей, можно выделить целое направление deep metric learning, которое в данной статье практически не затрагивалось.

## Список литературы

- [1] J. L. Suárez-Díaz, S. García, and F. Herrera, “A tutorial on distance metric learning: Mathematical foundations, algorithms, experimental analysis, prospects and challenges (with appendices on mathematical background and detailed algorithms explanation),” 2018.
- [2] K. Q. Weinberger, J. Blitzer, and L. Saul, “Distance metric learning for large margin nearest neighbor classification,” in *Advances in Neural Information Processing Systems* (Y. Weiss, B. Schölkopf, and J. Platt, eds.), vol. 18, MIT Press, 2005.
- [3] J. Goldberger, G. E. Hinton, S. Roweis, and R. R. Salakhutdinov, “Neighbourhood components analysis,” in *Advances in Neural Information Processing Systems* (L. Saul, Y. Weiss, and L. Bottou, eds.), vol. 17, MIT Press, 2004.
- [4] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, “Information-theoretic metric learning,” in *Proceedings of the 24th International Conference on Machine Learning, ICML '07*, (New York, NY, USA), p. 209–216, Association for Computing Machinery, 2007.
- [5] F. Wang and C. Zhang, “Feature extraction by maximizing the average neighborhood margin,” in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.
- [6] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka, “Metric learning for large scale image classification: Generalizing to new classes at near-zero cost,” in *Computer Vision – ECCV 2012* (A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, eds.), (Berlin, Heidelberg), pp. 488–501, Springer Berlin Heidelberg, 2012.
- [7] A. Globerson and S. Roweis, “Metric learning by collapsing classes.,” vol. 18, 01 2005.
- [8] Z. Huang, R. Wang, S. Shan, X. Li, and X. Chen, “Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification,” in *ICML*, 2015.
- [9] X. Pennec, P. Fillard, and N. Ayache, “A riemannian framework for tensor computing.,” *International Journal of Computer Vision*, vol. 66, no. 1, pp. 41–66, 2006.
- [10] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, “Geometric means in a novel vector space structure on symmetric positive-definite matrices,” *SIAM Journal on Matrix Analysis and Applications*, vol. 29, no. 1, pp. 328–347, 2007.

- [11] R. Wang, H. Guo, L. S. Davis, and Q. Dai, “Covariance discriminative learning: A natural and efficient approach to image set classification,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2496–2503, 2012.
- [12] M. Ha Quang, M. San Biagio, and V. Murino, “Log-hilbert-schmidt metric between positive definite operators on hilbert spaces,” in *Advances in Neural Information Processing Systems* (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds.), vol. 27, Curran Associates, Inc., 2014.
- [13] J. Hamm and D. Lee, “Extended grassmann kernels for subspace-based learning,” in *Advances in Neural Information Processing Systems* (D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, eds.), vol. 21, Curran Associates, Inc., 2008.
- [14] J. Hamm and D. D. Lee, “Grassmann discriminant analysis: A unifying view on subspace-based learning,” in *Proceedings of the 25th International Conference on Machine Learning, ICML ’08*, (New York, NY, USA), p. 376–383, Association for Computing Machinery, 2008.
- [15] Z. Huang, R. Wang, S. Shan, and X. Chen, “Projection metric learning on grassmann manifold with application to video based face recognition,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 140–149, 2015.
- [16] P. H. Zadeh, R. Hosseini, and S. Sra, “Geometric mean metric learning,” 2016.
- [17] P. Zhu, H. Cheng, Q. Hu, Q. Wang, and C. Zhang, “Towards generalized and efficient metric learning on riemannian manifold,” in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pp. 3235–3241, International Joint Conferences on Artificial Intelligence Organization, 7 2018.
- [18] S. Chen, C. Gong, J. Yang, X. Li, Y. Wei, and J. Li, “Adversarial metric learning,” in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pp. 2021–2027, International Joint Conferences on Artificial Intelligence Organization, 7 2018.
- [19] y. Zhang and D.-Y. Yeung, “Transfer metric learning by learning task relationships,” pp. 1199–1208, 11 2010.

- [20] J. Hu, J. Lu, and Y.-P. Tan, “Deep transfer metric learning,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 325–333, 2015.
- [21] Y. Xu, S. J. Pan, H. Xiong, Q. Wu, R. Luo, H. Min, and H. Song, “A unified framework for metric transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, pp. 1158–1171, 2017.
- [22] B. Cao, X. Ni, J.-T. Sun, G. Wang, and Q. Yang, “Distance metric learning under covariate shift,” in *IJCAI*, pp. 1204–1210, 2011.
- [23] Y. Ying and P. Li, “Distance metric learning with eigenvalue optimization,” *Journal of Machine Learning Research*, vol. 13, no. 1, pp. 1–26, 2012.
- [24] E. Hazan and S. Kale, “Projection-free online learning,” *CoRR*, vol. abs/1206.4657, 2012.
- [25] B. Nguyen, C. Morell, and B. De Baets, “Scalable large-margin distance metric learning using stochastic gradient descent,” *IEEE Transactions on Cybernetics*, vol. 50, no. 3, pp. 1072–1083, 2020.
- [26] H.-J. Ye, D.-C. Zhan, Y. Jiang, and Z.-H. Zhou, “What makes objects similar: A unified multi-metric learning approach,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 5, pp. 1257–1270, 2019.
- [27] S. Chen, L. Luo, J. Yang, C. Gong, J. Li, and H. Huang, “Curvilinear distance metric learning,” in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.