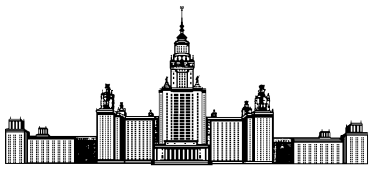
Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной Математики и Кибернетики

Кафедра Математических Методов Прогнозирования

Обзорная статья по теме «Metric Learning»

Выполнил:

студент 3 курса 317 группы

Борисов Алексей Антонович

Москва, 2022

Содержание

1 Введение

В общем виде задача 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 $\mathcal{X} = \{x_1, \ldots, x_N\}$ — 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 $\mathscr{L}$. Таким образом получаем следующую задачу оптимизации:

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

Если в дополнение к множеству $\mathcal{X}$ дан набор соответствующих меток $y_1, \ldots, y_N$, 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 metric learning 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 $y_1, \ldots, y_N$ задача оптимизации принимает следующий вид:

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

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 $M = L^T L$ и 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 $x \to Lx$, 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 соотвествует 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.

Несколько слов о структуре данной статьи. Сперва будет описано несколько классических методов metric learning, далее .... Также стоит сказать, что данная статья практически не будет касаться использования metric learning в глубоком обучении (deep metric learning), хотя многие современные статьи, связанные с metric learning, рассказывают о его применении в глубоком обучении.

2 Классические методы

Прежде чем переходить к более новым исследованиям в области metric learning, 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 допускать, чтобы объекты других классов не 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:

$$\varepsilon_{pull}(M) = \sum_{i=1}^{N} \sum_{j \rightsquigarrow i} d_M^2(x_i, x_j)$$

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

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

Функция потерь принимает вид:

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

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 невыпуклая.

Алгоритм LMNN можно использовать несколько раз, каждый раз получая новую метрику, так как после применения LMNN для каждого объекта можно выбрать более качественное

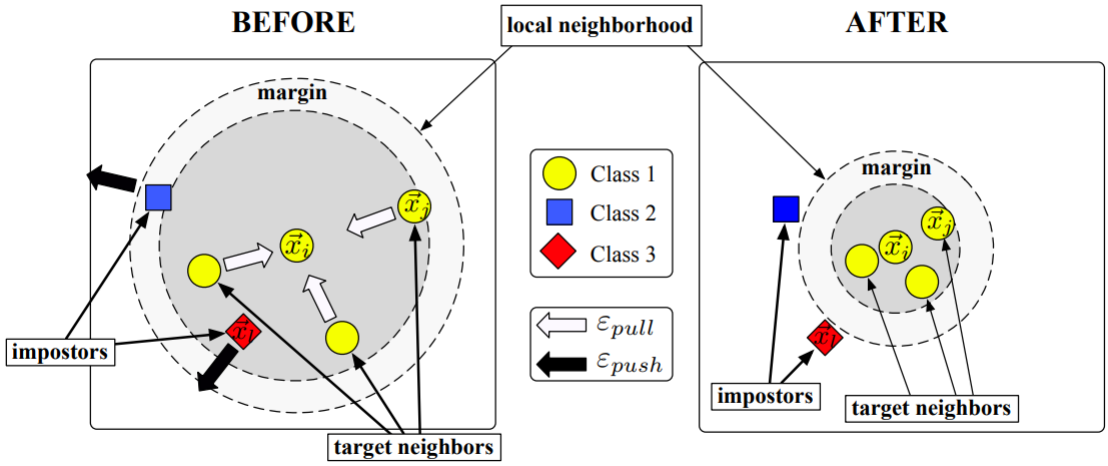


Рис. 1: Основная идея алгоритма LMNN

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:

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

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

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

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

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

2.3 Information-Theoretic Metric Learning

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

$$d_M(x_i, x_j) \leq u,\ (x_1, x_j) \in S$$
$$d_M(x_i, x_j) \geq l,\ (x_1, x_j) \in D$$

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 $\boldsymbol{\mu}$). Так, расстоянию Махаланобиса с матрицей M соответствует многомерное нормальное распределение:

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

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

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

Таким образом, оптимизационная задача для метода ITML принимает вид:

$$\min_M KL(p(\boldsymbol{x}; M_0) \| p(\boldsymbol{x}; M))$$

при условии $d_M(x_i, x_j) \leq u,\ (x_1, x_j) \in S$ и $d_M(x_i, x_j) \geq l,\ (x_1, x_j) \in D$

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

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

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

Так, задача сводится к следующей оптимизации LogDet дивергенции:

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

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

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

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 Metric learning на Римановом многообразии

Можно выделить целое направление работ по metric learning, в которых расстояние ищется 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 Metric learning в пространстве положительно определенных матриц

Пусть $\{X_1, \ldots, X_n\}, X_i \in \mathbb{R}^{d \times m_i}$, где d — размерность пространства признаков, а m_i — количество объектов. X_i соответствует значение целевой переменной l_i. В качестве X_i может выступать набор фотографий или множество локальных признаков, извлеченных из изображения. По $\{X_1, \ldots, X_n\}$ может быть сформирован набор положительно определенных матриц $\{S_1, \ldots, S_n\}, S_i \in \mathbb{R}^{d \times d}$, 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:

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

где $\boldsymbol{m}$ — среднее, а $\boldsymbol{S}$ — 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 считается расстояние, порожденное нормой Фробениуса.

Было создано большое количество подходов для использования metric learning в пространстве положительно определенных матриц, например, Affine Invariant Riemannian Metric (AIRM), Log-Euclidean Riemannian Metric (LERM). 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.

3.2 Metric learning 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 Λ_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 $\mathbb{R}^d$, позволяющем разделять объекты разных классов. Однако, в статье [9] обучается отображение в другое многообразие Грассмана, в котором объекты разных классов лучше разделяются. Иллюстрация обоих подходов изображена на рисунке 2.

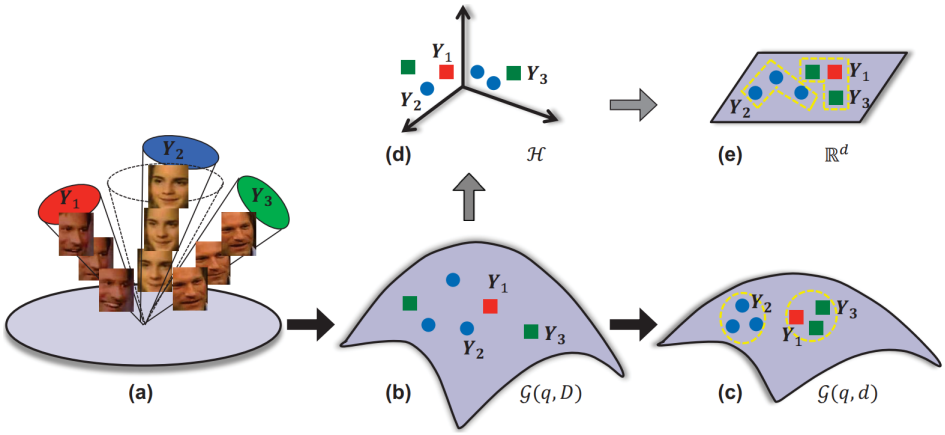


Рис. 2: Иллюстрация основной идеи методов metric learning, использующих многообразие Грассмана. (a)-(b)-(d)-(e) представляет собой первый вариант, с отображением в гильбертово пространство $\mathcal{H}$, а (a)-(b)-(c) вариант с отображением в другое многообразие Грассмана.

3.3 Geometric Mean Metric Learning

Очень интересной является следующая статья [10]. 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 $\boldsymbol{M}$ преследует следующие две идеи: расстояние между похожими объектами в метрике, порожденной матрицей $\boldsymbol{M}$, нужно минимизировать, а расстояние между непохожими объектами нужно

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

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

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

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

После введения следующих обозначений:

$$\boldsymbol{S} = \sum_{(x_i,x_j)\in S} (x_i - x_j)(x_i - x_j)^T,$$
$$\boldsymbol{D} = \sum_{(x_i,x_j)\in D} (x_i - x_j)(x_i - x_j)^T,$$

оптимизационную задачу можно записать в следующем виде:

$$\min_{\boldsymbol{M}\succeq 0} \mathscr{L}(\boldsymbol{M}) = tr(\boldsymbol{M}\boldsymbol{S}) + tr(\boldsymbol{M}^{-1}\boldsymbol{D})$$

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: $\boldsymbol{M}\boldsymbol{S}\boldsymbol{M} = \boldsymbol{D}$. Единственным решением этого уравнения является середина геодезической кривой, соединяющей $\boldsymbol{S}^{-1}$ и $\boldsymbol{D}$, которая определяется следующим образом: $\boldsymbol{A}\sharp_t\boldsymbol{B} = \boldsymbol{A}^{1/2}(\boldsymbol{A}^{-1/2}\boldsymbol{B}\boldsymbol{A}^{-1/2})^t\boldsymbol{A}^{1/2},\ t \in [0,1]$.

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

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

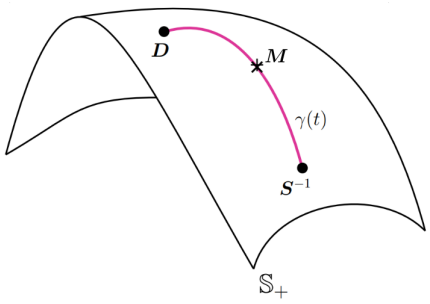


Рис. 3: Решение задачи GMML расположено в середине геодезической кривой, соединяющей атрицы $\boldsymbol{S^{-1}}$ и $\boldsymbol{D}$ в пространстве положительно определенных матриц.

3.4 Обобщенный подход

Как можно заметить, методы metric learning в пространстве положительно определенных матриц и на многообразии Грассмана имеют много общего. Так в статье [11], 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:

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

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

А для многообразия Грассмана:

$$\boldsymbol{S} = \sum_{(x_i,x_j)\in S} \boldsymbol{T}_{ij}\boldsymbol{T}_{ij}, \qquad \boldsymbol{D} = \sum_{(x_i,x_j)\in D} \boldsymbol{T}_{ij}\boldsymbol{T}_{ij}, \qquad \text{где } \boldsymbol{T}_{ij} = Y_iY_i^T - Y_jY_j^T.$$

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

$$\min_{\boldsymbol{M}\succeq 0} \mathscr{L}(\boldsymbol{M}) = tr(\boldsymbol{M}\boldsymbol{S}) + tr(\boldsymbol{M}^{-1}\boldsymbol{D})$$

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

$$\boldsymbol{M}_{final} = \exp\Big(\frac{-t\log(\boldsymbol{S}+\lambda\boldsymbol{I}) + (1-t)\log(\boldsymbol{D}+\lambda\boldsymbol{I})}{2}\Big)$$

4 Применение современных техник

В этом разделе речь пойдет об использовании в metric learning подходов, хорошо зарекомендовавших себя в других областях машинного обучения.

4.1 Adversarial Metric Learning

В сатье [12] для обеспечения лучшей робастности и дискриминативной способности предлагается использовать состязательное обучение (adversarial training).

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

В отличии от классических методов metric learning алгоритм AML состоит из двух этапов: *confusion* и *distinguishment*.

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

$$\min_{\boldsymbol{\Pi}} C_{\boldsymbol{M}}(\boldsymbol{\Pi}) = \mathscr{L}(\boldsymbol{M}, \boldsymbol{\Pi}, -\boldsymbol{y}) - \beta \mathrm{Dist}^{+}_{\boldsymbol{M}}(\boldsymbol{X}, \boldsymbol{\Pi}),$$

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

На рисунке 4 показано, каким образом генерируются новые пары.

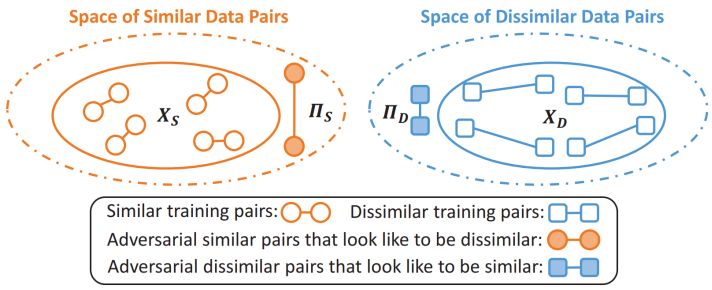


Рис. 4: Процесс генерации новых пар в методе AML.

В свою очередь на этапе *distinguishment* алгоритм должен обновить метрику $\boldsymbol{M}$ 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:

$$\min_{\boldsymbol{M} \succeq 0} D_{\boldsymbol{\Pi}}(\boldsymbol{M}) = \mathscr{L}(\boldsymbol{M}, \boldsymbol{X}, \boldsymbol{y}) + \alpha \mathscr{L}(\boldsymbol{M}, \boldsymbol{\Pi}, \boldsymbol{y})$$

Коэффициенты α и β определяют влияние соответствующих слагаемых при оптимизации.

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

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

Однако для получившейся задачи не гарантируется сходимость, и авторы статьи свели ее похожей задаче, для которой гарантируется сходимость:

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

При этом функция $C_{\boldsymbol{M}}(\boldsymbol{\Pi})$ должна быть строго квазивыпуклой, что гарантирует единственность $\boldsymbol{\Pi}^*$. Для этого авторы использовали функцию потерь, которая использовалась в GMML. 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) 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) [13], 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.

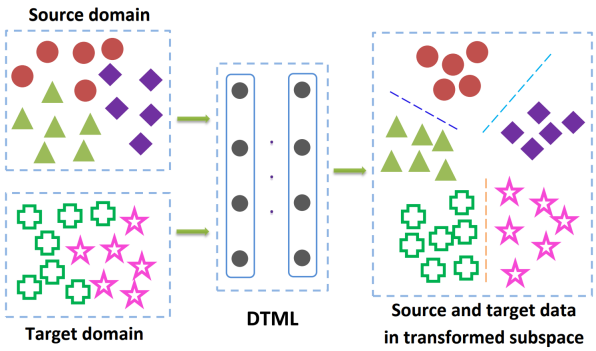


Рис. 5: Основная идея алгоритма DTML.

В более современном подходе, который авторы назвали Metric Transfer Learning Framework (MTLF) [14], представлен более совершенный подход, который заключается в одновременном обучении как весов для объектов обучающей выборки, так и метрики Махаланобиса. Эта особенность является одним из основних отличий этого метода от более старого метода consistent distance metric learning (CDML), в котором сначала обучались веса объектов под использование с евклидовой метрикой, а затем на перевзвешенных объектах обучалась метрика.

5 Некоторые проблемы metric learning

Как было показано выше, многие методы metric learning 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.

Методы metric learning зачастую обучаются на парах или тройках объектов, поэтому с ростом N количество обучающих примеров очень быстро растет. Для решения этой проблемы используют онлайновые методы обучения, например, градиентный спуск. 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.

TODO

5.1 Scalable Distance Metric Learning

5.2 Metric Learning в случае дисбаланса классов

6 Нелинейные подходы в metric learning

6.1 Multi-metric learning

6.2 Local metric learning

6.3 Curvilinear metric learning

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] 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.

[10] P. H. Zadeh, R. Hosseini, and S. Sra, "Geometric mean metric learning," 2016.

[11] 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.

[12] 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.

[13] 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.

[14] 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.