ДОКЛАДЫ РОССИЙСКОЙ АКАДЕМИИ НАУК. МАТЕМАТИКА, ИНФОРМАТИКА, ПРОЦЕССЫ УПРАВЛЕНИЯ, 2023, том 514, № 2, с. 60–71

УДК 519.7

НЕЙРОСЕТЕВОЕ ОБУЧЕНИЕ МЕТРИК: СРАВНЕНИЕ ФУНКЦИЙ ПОТЕРЬ

© 2023 г. Р. Л. Васильев[1,*], А. Г. Дьяконов[2,]**

Представлено академиком РАН А.А. Шананиным
Поступило 30.06.2023 г.
После доработки 19.09.2023 г.
Принято к публикации 15.10.2023 г.

Представлен обзор методов обучения метрик с помощью глубоких нейронных сетей. Эти методы появились в последние годы, но сравнивались лишь с предшественниками, используя для обучения представлений (на которых вычисляется метрика) нейронные сети устаревших на данный момент архитектур. Проведено сравнение описанных методов на разных датасетах из нескольких доменов, используя предобученные нейронные сети, сопоставимые по качеству с SotA (state of the art): ConvNeXt для изображений, DistilBERT для текстов. Использовались размеченные наборы данных, разбитые на две части (обучение и контроль) таким образом, чтобы классы не пересекались (т.е. в контроле нет объектов тех классов, которые были в обучении). Подобное масштабное честное сравнение сделано впервые и привело к неожиданным выводам: некоторые "старые" методы, например Tuplet Margin Loss, превосходят по качеству свои современные модификации и методы, предложенные в совсем свежих работах.

Ключевые слова: машинное обучение, глубокое обучение, метрика, схожесть

DOI: 10.31857/S268695432360060X, **EDN:** IDAFOE

1. ВВЕДЕНИЕ

Во множестве задач анализа данных и машинного обучения возникает потребность в оценке близости объектов: в поиске по изображениям [1] или текстовым документам [2], распознавании [3] и идентификации [4] лиц. Современные методы решения таких задач используют *нейросетевое обучение метрик* (Deep Metric Learning). Предполагается, что "обучить метрику" значит получить некоторый автоматический способ вычисления расстояний (например, в виде выхода нейросети), при котором расстояния между объектами, которые должны быть похожи (например, лежат в одном классе), малы, а расстояния между объектами, которые должны различаться – велики.

Опишем эту идею более формально. Рассмотрим множество объектов X и множество меток Y. Объекты $x \in X$ могут иметь произвольную природу (изображения, тексты, аудио, табличные данные и т.д.), каждому объекту может соответствовать некоторая метка $y \in Y$. Если объекты индексированы, то считаем, что объекту x_t соответствует метка y_t. Общая задача *обучения метрики* (Metric Learning) – построить метрику $D_\theta(\cdot, \cdot) : X \times X \to \mathbb{R}$ из параметрического семейства $\{D_\theta\}$, удовлетворяющую свойству

$$D_\theta(x_1, x_2) \ll D_\theta(x_1, x_3)$$

при $y_1 = y_2 \neq y_3$. Не во всех задачах есть метки объектов, но часто есть способ построить пару (x_1, x_2) объектов, которые логично считать похожими, например объект и его аугментация:

- сигнал и его зашумленная копия;
- изображение и его повернутая копия;
- текст и такой же текст, в котором некоторые слова заменены синонимами. Такие пары называются *позитивными*, остальные – *негативными*, например случайная пара объектов выборки. Формально, для множества позитивных пар P и негативных пар N можно выписать систему неравенств

$$D_\theta(x_i, x_j) < D_\theta(x_t, x_s), \quad (i, j) \in P, \quad (t, s) \in N,$$

но в такой системе большое число неравенств, кроме того, знак "$\ll$" в описанном нами свойстве означает желание того, чтобы эти расстояния существенно отличались, поэтому применяют подходы, которые мы опишем ниже.

[1] *ООО "Яндекс", Москва, Россия*
[2] *Центральный университет, Москва, Россия*
**E-mail: artnitolog@yandex.com*
***E-mail: djakonov@mail.ru*

В задаче *нейросетевого обучения метрик* (Deep Metric Learning) требуется обучить нейронную сеть – отображение $f_\theta(\cdot) : X \to \mathbb{R}^n$, при этом

$$D_\theta(x_1, x_2) = D(f_\theta(x_1), f_\theta(x_2)),$$

где D – фиксированная простая метрика в конечномерном пространстве $\mathbb{R}^n$ векторных представлений (embeddings) объектов, чаще используют евклидово расстояние. Также часто ограничивают параметризацию единичной сферой: $\|f_\theta(x)\| = 1$ и вводят не $D_\theta(x_1, x_2)$, а скалярное произведение:

$$S_\theta(x_1, x_2) = \langle f_\theta(x_1), f_\theta(x_2)\rangle,$$

здесь уже стремятся к выполнению

$$S_\theta(x_1, x_2) \gg S_\theta(x_1, x_3)$$

при $y_1 = y_2 \neq y_3$.

Для поиска оптимальных параметров θ задается функция потерь L и оптимизируется на обучающей выборке. Далее в работе рассмотрим конкретные функции потерь.

Обучение метрик полезно в задачах с разметкой (Supervised Learning), когда классов довольно много и они малопредставлены. Также когда не все классы есть в обучающей выборке и алгоритм должен детектировать появление объектов из неизвестных классов (Open-World Classification).

В данной работе принимается попытка систематизировать нейросетевые подходы к обучению метрик, делается большое число экспериментов с целью сравнить разные функции потерь в разных доменах: изображения и тексты. Подобное масштабное сравнение сделано впервые. В разд. 2 рассматриваются основные функционалы качества, позволяющие сравнить методы друг с другом. В разд. 3 приводится обзор различных методов: основанных на сравнении представлений или на сведении к задаче классификации. В разд. 4 описываются эксперименты с данными методами на разных датасетах, нейросетевых архитектурах и модальностях.

2. ПОКАЗАТЕЛИ КАЧЕСТВА

2.1. Поисковые

Назовем *запросом* (query) объект, для которого производится поиск ближайших соседей, а объекты, среди которых производится поиск – *документами* (documents); терминология пришла из информационного поиска. По запросу алгоритм информационного поиска выдает упорядоченный список документов, для этого как раз можно использовать обученную метрику и упорядочивать документы по возрастанию расстояния от запроса. При этом среди всех документов есть подмножество *релевантных* документов – тех, которые хотелось бы выдавать, причем они должны располагаться вверху нашего перечня. Следующие показатели качества как раз оценивают, насколько это получается.

Один из наиболее часто используемых показателей качества в обучении метрики – Recall@K, который определяется как доля запросов, для которых среди K ближайших соседей нашелся релевантный документ. Отметим, что такое определение Recall@K используется именно в задачах обучения метрики, в информационном поиске под Recall@K обычно понимается доля найденных объектов среди релевантных.

Другой часто используемый показатель качества – MAP (Mean Average Precision). Определим Precision@K как долю релевантных объектов среди K найденных. Пусть в выдаче индексы релевантных объектов: $K_1, \ldots, K_R$, тогда можно вычислить

$$\mathrm{AP} = \frac{1}{R}\sum_{i=1}^{R}\mathrm{Precision@}K_i$$

и MAP – это усреднение AP по всем запросам.

Зачастую также считают R-Precision, равную Precision@R, где R – общее число релевантных документов. В [5] предлагается функционал, объединяющий идеи R-Precision и MAP. Для отдельного запроса с R релевантными документами он равен

$$\mathrm{AP@R} = \frac{1}{R}\sum_{i=1}^{R}\mathrm{Precision@i},$$

усреднением по всем запросам получается общая MAP@R для всего датасета.

Также измеряют *среднеобратный ранг* (MRR – Mean Reciprocal Rank):

$$MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\mathrm{rank}_i},$$

где rank_i – порядковый номер первого релевантного документа, m – общее число запросов.

2.2. Кластерные

Кроме поисковых критериев качества в обучении метрики нередко оценивается качество кластеризации в пространстве $\mathbb{R}^n$ – представлений $\{f_\theta(x)|x \in X\}$: ожидается, что представления похожих объектов будут образовывать кластеры. Недостаток основанных на этой идеи “кластерных” функционалов качества – зависимость функционалов от выбранного алгоритма кластеризации.

Пусть $U = \{U_1, \ldots, U_n\}$ – истинное разбиение X на группы похожих объектов, а $V = \{V_1, \ldots, V_n\}$ – полученное алгоритмом кластеризации (в данной

работе в экспериментах используется k-means [6]) на $f_\theta(X)$. Тогда *взаимной информацией* (mutual information) называется

$$MI(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} P_{UV}(i,j)\log\frac{P_{UV}(i,j)}{P_U(i)P_V(j)} = KL(P_{UV} \parallel P_U P_V),$$

где $P_U(i) = \dfrac{|U_i|}{|U|}$ – оценка вероятности случайного объекта попасть в i-й кластер разбиения U, аналогично $P_V(i) = \dfrac{|V_i|}{|V|}$ и $P_{UV}(i,j) = \dfrac{|U_i \cap V_j|}{|V|}$ – оценка вероятности попасть одновременно в U_i и V_j.

Энтропия разбиения S определяется следующим образом:

$$H(S) = -\sum_{k=1}^{|S|} P_S(k)\log P_S(k).$$

Существует несколько нормализаций: NMI (normalized mutual information) и AMI (adjusted mutual information). NMI вычисляется следующим образом:

$$NMI(U,V) = \frac{MI(U,V)}{\frac{1}{2}(H(U)+H(V))},$$

но чаще используют AMI:

$$AMI(U,V) = \frac{MI(U,V) - \mathbb{E}MI(U,V)}{\frac{1}{2}(H(U)+H(V)) - \mathbb{E}MI(U,V)},$$

где $\mathbb{E}MI(U,V)$ – матожидание $MI(U,V)$ по всем возможным разбиениями U и V.

3. МЕТОДЫ НЕЙРОСЕТЕВОГО ОБУЧЕНИЯ МЕТРИКИ

3.1. Методы, основанные на сравнении представлений

Опишем методы, которые используются для обучения метрики и основаны на сравнении представлений объектов в пространстве значений $\{f_\theta(x)|x \in X\}$. Каждый метод определяется функцией потерь, которая используется при оптимизации нейронной сети, которая задает отображение $f_\theta(x)$.

3.1.1. Contrastive Loss. Одной из первых функций потерь, предложенных для обучения метрики, была *Contrastive Loss* [7], которая определяется следующим образом:

$$L = \mathbb{I}\{y_i = y_j\}\left[D_\theta(x_i,x_j) - m_p\right]_+^2 + \mathbb{I}\{y_i \neq y_j\}\left[m_n - D_\theta(x_i,x_j)\right]_+^2, \quad (1)$$

здесь и далее $\mathbb{I}\{A\} = 1$ тогда и только тогда, когда выражение A истинно, $[z]_+ = \max\{z,0\}$ – функция срезки. Функционал (1) штрафует позитивные пары, если расстояние между ними больше некоторого порога m_p (чаще всего полагают равным нулю), а негативные – до тех пор, пока расстояние не превысит порог m_n.

3.1.2. Triplet Loss. В [8] предложили сравнивать не пары, а тройки объектов (x_a, x_p, x_n), где $y_a = y_p$, но $y_a \neq y_n$:

$$L = [D_\theta(x_a,x_p)^2 - D_\theta(x_a,x_n)^2 + m]_+. \quad (2)$$

Основное отличие (1) от (2): Contrastive Loss штрафует абсолютное расстояние в рамках одной пары объектов, в то время как Triplet Loss ограничивает *разницу* между *позитивными* и *негативными* парами в тройке. Triplet Loss зачастую применяется в задаче распознавания лиц [8], а его модификации используются и в других областях, например, для обучения текстовых представлений [2].

3.1.3. Fast AP. Как было замечено в разд. 2, одним из основных функционалов качества в задачах Metric Learning является MAP. В [9] предлагается оптимизировать аппроксимацию AP. Основная проблема функционала с точки зрения оптимизации – операция сортировки, для которой нельзя применить градиентные методы. Идея авторов – интерпретировать AP как площадь под PR-кривой и рассмотреть Precision и Recall как параметрические функции от расстояния между запросами и объектами. Было предложено следующее приближение:

$$\text{FastAP} = \frac{1}{N_q^+}\sum_{j=1}^{L}\frac{H_j^+ h_j^+}{H_j}. \quad (3)$$

• Предполагается, что представления l_2-нормализованы: $\|f_\theta(x)\| = 1$, – в этом случае возможные расстояния между запросом query и кандидатами retrieval принадлежат отрезку [0, 2].

• Отрезок [0, 2] разбивается на бины $\{z_1,\ldots,z_L\}$ (количество бинов L – гиперпараметр), и h_j – число объектов, попавших в j-й бин. $H_j = \sum_{k=1}^{j} h_k$ – кумулятивная сумма.

• h_j^+ и H_j^+ – те же счетчики, но только для релевантных запросу объектов; N_q^+ – общее число релевантных объектов.

• Фактически в (3) используется *гистограммный биннинг* histogram binning, который в данном случае приближает истинное распределение (а именно, плотность и функцию распределения) расстояний кусочно-постоянными функциями:

$$h(z) = \sum_{j=1}^{L} h_j \cdot \mathbb{I}\{z \in z_j\}, \quad H(z) = \sum_{j=1}^{L} H_j \cdot \mathbb{I}\{z \in z_j\}.$$

• Поскольку *кусочно-постоянные функции* недифференцируемы, при оптимизации вместо $h(z)$ и $H(z)$ используется линейная интерполяция, приводящая к непрерывным *кусочно-линейным функциям* – их градиенты определены и постоянны внутри отдельных бинов. Такая релаксация была предложена в [10].

3.1.4. Centroid Triplet Loss. В [11] авторы предлагают модифицировать (2), заменив позитивные и негативные объекты (x_p, x_n) относительно x_a на центры их классов – (c_a, c_p):

$$L = [D_\theta(x_a, c_p)^2 - D_\theta(x_a, c_n)^2 + m]_+.$$

Во время обучения центры классов считаются по батчу (а сами батчи стоит делать большими), причем x_a исключается. Основная мотивация данного метода – упрощение этапа применения: поиск можно производить не по всем объектам, а только по центрам, которые предподсчитываются заранее (уже по всей выборке).

3.1.5. Margin Loss. Авторы [12] предлагают использовать

$$L = \left[\alpha + \left(2 \cdot \mathbb{I}\{y_i = y_j\} - 1\right) \cdot \left(D_\theta(x_i, x_j) - \beta\right)\right]_+. \quad (4)$$

Фактически (4) отличается от (1) заменой квадрата евклидовой метрики на саму метрику $(l_2^2 \mapsto l_2)$ и измененной параметризацией: $m_p = \beta - \alpha$, $m_n = \beta + \alpha$. Таким образом, β соответствует границе между *позитивными* и *негативными* парами, а α – необходимому отступу от этой границы.

3.1.6. Multi Similarity Loss. В [13] в функции потерь предлагается использовать больше информации о расстояниях между объектами в батче:

$$L = \sum_{i=1}^{m} \left\{ \frac{1}{\alpha} \log\left[1 + \sum_{k \in P_i} e^{-\alpha(S_{ik} - \lambda)}\right] + \frac{1}{\beta} \log\left[1 + \sum_{k \in N_i} e^{\beta(S_{ik} - \lambda)}\right] \right\}. \quad (5)$$

Здесь параметризуется $S_\theta(x_i, x_j) = \langle f_\theta(x_i), f_\theta(x_j) \rangle$. Отметим, что выражение (5) можно рассматривать как гладкую аппроксимацию (1) (с дополнительными коэффициентами), включающую все попарные расстояния внутри батча.

3.1.7. SNN Loss. Функцию потерь на основе расстояний внутри батча можно определить и следующим образом [14]:

$$L = -\frac{1}{m} \sum_{i=1}^{m} \log \frac{\sum_{j=1}^{m} \mathbb{I}\{y_i = y_j\} \exp\{S_\theta(x_i, x_j)/\tau\}}{\sum_{j=1}^{m} \exp\{S_\theta(x_i, x_j)/\tau\}}, \quad (6)$$

τ – температура (может быть как настраиваемым параметром, так и гиперпараметром).

3.1.8. SupCon Loss. В [15] авторы заметили, что в (6) позитивные примеры можно агрегировать по-разному, и предложили следующую альтернативу:

$$L = -\frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{m} \mathbb{I}\{y_i = y_j\} \log \frac{\exp\{S_\theta(x_i, x_j)/\tau\}}{\sum_{j=1}^{m} \exp\{S_\theta(x_i, x_j)/\tau\}}. \quad (7)$$

Вариант (7) обычно оптимизируется лучше (6).

3.1.9. SNR Loss. В [16] используется (1), но вместо евклидовой метрики авторы оптимизируют SNR (signal-to-noise ratio):

$$D_\theta(x_i, x_j) = \frac{1}{\mathrm{SNR}} = \frac{\mathrm{Var}\left[f_\theta(x_i) - f_\theta(x_j)\right]}{\mathrm{Var}\left[f_\theta(x_i)\right]},$$

здесь Var – выборочная дисперсия. В отличие от евклидовой метрики, SNR-расстояние не симметрично, поэтому порядок элементов в паре имеет значение, аналогично тому, как на Triplet Loss (2) влияет замена $x_a \leftrightarrow x_p$.

3.1.10. Tuplet Margin Loss. Авторы [17] внесли несколько существенных изменений в (2):

$$L = \log\left(1 + \sum_{i=1}^{k-1} e^{s(\cos\theta_{a,n_i} - \cos(\theta_{a,p} - \beta))}\right). \quad (8)$$

• В (8) число негативных объектов не ограничено триплетом: для каждой позитивной пары (x_a, x_p) из оставшихся классов семплируется по 1 негативному примеру, формируя набор

$$(x_a, x_p, x_{n_1}, \ldots, x_{n_{k-1}}).$$

• Используются представления на сфере $(\|f_\theta(x_i)\| = 1)$, поэтому косинус угла между векторами вычисляется как скалярное произведение.

• Коэффициент $\beta \geqslant 0$ используется для борьбы с переобучением на hard triplets (триплеты, в которых $\theta_{a,n_i} \ll \theta_{a,p}$), которые с $\beta = 0$ вносят в функцию потерь сильно больший вклад относительно других элементов набора.

• Вместо функции срезки $[z]_+ = \max\{z, 0\}$ используется ее дифференцируемая аппроксимация $\mathrm{Softplus}(z) = \log(1 + e^{s \cdot z})$, где s – коэффициент масштаба (отвечает за радиус гиперсферы и влияет на скорость сходимости).

3.1.11. Circle Loss. В [18] предлагается оптимизировать модифицированную релаксацию (1):

$$L = \log\left[1 + \sum_{j=1}^{L} e^{\gamma[s_n^j - m_n]_+ s_n^j} \sum_{i=1}^{K} e^{-\gamma[m_p - s_p^i]_+ s_p^i}\right], \qquad (9)$$

где s_p^j и s_n^i – функционалы сходства между позитивными и негативными парами.

3.2. Методы, основанные на классификации

Данная группа методов отличается тем, что для обучения используется выборка, размеченная на фиксированное число классов c. Фактически решается задача классификации, но стандартные методы классификации не гарантируют получение пространства представлений, в котором объекты каждого класса хорошо группируются.

3.2.1. ArcFace. В ArcFace [19] предлагается функция потерь, оптимизируя которую, авторам удалось получить качественные векторные представления в задаче распознавания лиц. Идея заключается в том, чтобы рассмотреть Softmax-loss на единичной сфере и использовать геодезическое расстояние.

Введем вспомогательную матрицу весов $W \in \mathbb{R}^{n\times c}$. При решении задачи классификации можно было бы оптимизировать Softmax-loss (кросс-энтропию):

$$-\log \frac{e^{W_{y_i}^T f_\theta(x_i)}}{\sum_{j=1}^{c} e^{W_j^T f_\theta(x_i)}}. \qquad (10)$$

Рассмотрим j-й логит (10): $W_j^T f_\theta(x_i) = \|W_j\|\,\|f_\theta(x_i)\|\cos_{\theta_j}$, где θ_j – угол между W_j и $f_\theta(x_i)$. Зафиксируем норму всех весов $\|W_j\| = 1$, а также нормализуем пространство представлений: $\|f_\theta(x_i)\| = s$, в этом случае (10) примет вид (11):

$$L = -\log \frac{e^{s(\cos(\theta_{y_i}))}}{e^{s(\cos(\theta_{y_i}))} + \sum_{j\neq y_i} e^{s\cos(\theta_j)}}, \qquad (11)$$

добавив смещение к углу в (11), получим ArcFace Loss [19]:

$$L = -\log \frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))} + \sum_{j\neq y_i} e^{s\cos(\theta_j)}}. \qquad (12)$$

Параметр m в данном случае отвечает и за внутриклассовую компактность, и за межклассовое расстояние.

3.2.2. CosFace. Практически та же самая идея была предложена в [20], но авторы использовали смещение не для углов, а для косинусов:

$$L = -\log \frac{e^{s(\cos(\theta_{y_i})+m)}}{e^{s(\cos(\theta_{y_i})+m)} + \sum_{j\neq y_i} e^{s\cos(\theta_j)}}. \qquad (13)$$

Хотя методы (12) и (13) нередко имеют практически одинаковое качество [19], в задаче распознавания лиц ArcFace часто работает чуть лучше.

3.2.3. SubCenter ArcFace. В [21] предлагается усилить устойчивость функционала (12) к шуму: для этого предлагается для каждого класса использовать K подцентров вместо одного. Вид функции потерь (12) прежний, но вводится K матриц весов $W^1,\ldots,W^K \in \mathbb{R}^{n\times c}$:

$$\theta_j = \arccos\left(\max_{k\in 1,\ldots,K}\{(W^k)_j^T f_\theta(x_i)\}\right). \qquad (14)$$

Таким образом, в (14) среди K подцентров выбирается один ближайший. При таком подходе, как показано в [21], большая часть объектов конкретного класса будет собираться в окрестности одного подцентра, в то время как шумовым объектам присвоятся другие *непопулярные* (содержащие малое число представителей) подцентры, которые можно будет отбросить на этапе применения.

3.2.4. SoftTriple Loss. В работе [22] также рассматривается идея с несколькими центрами для одного класса.

$$L = -\log \frac{e^{s(G(i,y_i)+m)}}{e^{s(G(i,y_i)+m)} + \sum_{j\neq y_i} e^{sG(i,j)}},$$

где G – функция близости между i-м объектом и классом c, w_c^k – k-й центр класса c:

$$G(i,c) = \sum_k \frac{\exp\left\{\frac{1}{\gamma}\langle f_\theta(x_i), w_c^k\rangle\right\}}{\sum_{k'}\exp\left\{\frac{1}{\gamma}\langle f_\theta(x_i), w_c^{k'}\rangle\right\}}\langle f_\theta(x_i), w_c^k\rangle.$$

3.2.5. Proxy-Anchor Loss. Авторы [23] предлагают добавлять специальные прокси-объекты оптимизировать расстояние между ними и объектами одного класса. Некоторые методы выше являются таковыми: например, в (12) центры классов θ_j можно рассматривать как прокси-объекты. В [23] используется следующая функция потерь:

Таблица 1. Разбиение на обучающую и контрольную выборки, классы между разбиениями не пересекаются

	Число классов в обучении	Число классов в контрольной выборке	Число объектов в обучении	Число объектов в контрольной выборке
Cars-196	98	98	8006	8179
CUB-200	100	100	5897	5891
SOP	11317	11317	59989	60064
Dogs-130	65	65	36904	33528
News-20	10	10	9622	9224
WOS-134	67	67	24283	22702

$$L = \frac{1}{|P^+|}\sum_{p\in P^+}\log\left(1+\sum_{x\in X_p^+}e^{-\alpha(S(x,p)-m)}\right)+$$
$$+\frac{1}{|P|}\sum_{p\in P}\log\left(1+\sum_{x\in X_p^-}e^{-\alpha(S(x,p)-m)}\right),$$

где α – гиперпараметр масштаба, m – сдвига, P – множество всех прокси-объектов, P^+ – положительных (в батче), X_p^-, X_p^+ и X_p^- – разбиение батча по прокси-объектам.

4. ЭКСПЕРИМЕНТЫ

Как мы видели в предыдущем разделе, существует довольно много функций потерь для решения задачи обучения метрики. Часто при выборе конкретного метода руководствуются наличием готовой реализации или популярностью метода. В данной работе было решено реализовать и протестировать все описанные методы, обеспечив честность оценки и сравнения их качества.

4.1. Данные

Одной из важнейших особенностей обучения метрики является сохранение свойств метрики (похожие объекты – близки, объекты разных классов – нет) на новых данных. Для честного измерения качества в Supervised-данных обучение и тестирование производится на непересекающихся классах.

В данной работе используется 6 датасетов: 4 с изображениями и 2 текстовых, объекты в них разделены на обучающую и контрольную выборки в соответствии с табл. 1.

4.1.1. Cars-196. Исходный датасет Cars-196 [24] содержит 16185 фотографий, разбитых на 196 классов. На каждой фотографии изображен автомобиль, причем класс характеризуется маркой, годом и моделью (цвет, ракурс, фон и т.д. могут отличаться). Примеры изображений приведены на рис. 1. Это один из наиболее часто использующихся датасетов в замерах качества методов Metric Learning.

4.1.2. CUB-200. В датасете Caltech-UCSD Birds 200 [25] содержится 11788 фотографий птиц, всего 200 категорий, примеры приведены на рис. 2.

4.1.3. SOP. Датасет Stanford Online Products [26] содержит 120053 изображений различных товаров, общее число категорий – 22634, примеры приведены на рис. 3.

4.1.4. Dogs-130. В датасете Tsinghua Dogs [27] всего 70432 различных фотографий собак, всего собрано 130 пород рис. 4.

4.1.5. News-20. Датасет 20 Newsgroups [28] обычно используется для сравнения методов классификации, в данной же работе он используется для построения текстовых представлений. Всего в датасете 18846 текстов и 20 классов.

4.1.6. WOS-134. В датасете Web Of Science [29] всего 46985 текстовых документов. Число категорий достаточно велико: 134, поэтому на нем подходы Deep Metric Learning могут быть особенно актуальны.

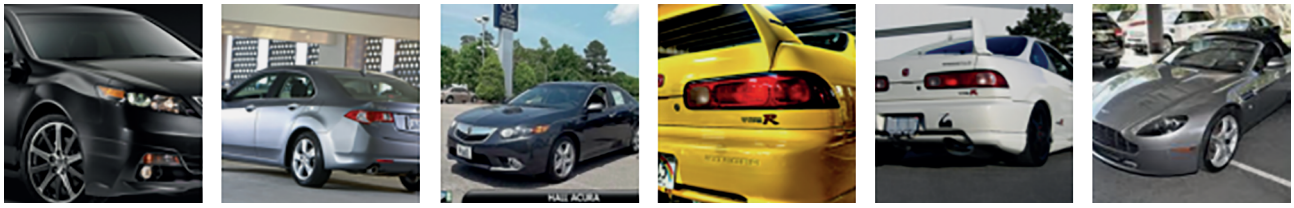

Рис. 1. Cars-196: случайная контрольная подвыборка после предобработки.

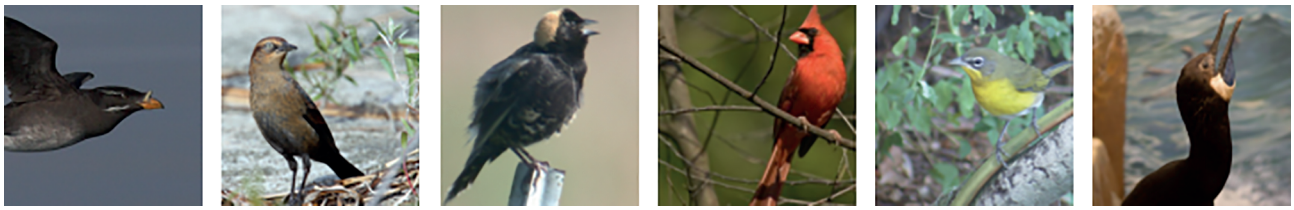

Рис. 2. CUB-200: случайная контрольная подвыборка после предобработки.

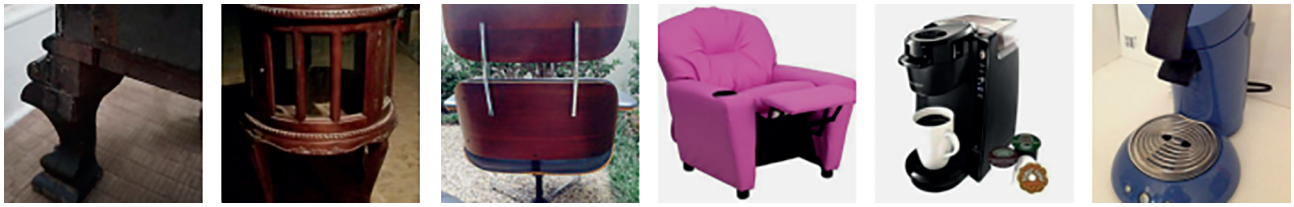

Рис. 3. SOP: случайная контрольная подвыборка после предобработки.

4.2. Нейросетевые архитектуры

4.2.1. ConvNeXt. Большая часть рассмотренных ранее методов в оригинальных статьях сравнивалась на архитектурах 2014–2015 гг. Resnet-50 [30] и GoogLeNet [31]. С тех пор появилось множество других моделей, для которых обучение метрик не исследовалось. В данной работе в качестве основной нейронной сети для изображений используется ConvNeXt [32], имеющая сверточную архитектуру, но сопоставимую по качеству с современными трансформерными моделями. Будем использовать предобученную ConvNeXt-T, имеющую 28M обучаемых параметров. Поверх нейронной сети достроим линейный слой, переводящий скрытое состояние в пространство нужной размерности.

4.2.2. DistilBERT. Для работы с текстовыми данными в данной работе используется трансформерная модель DistilBERT [33] – дистиллированная версия BERT [34]. Для получения векторных представлений необходимой размерности также достроим линейный слой поверх модели, конкретно – CLS-токена.

4.2.3. Детали обучения. Основные использовавшиеся фреймворки для написания скриптов обучения и замеров – PyTorch и [35]. В качестве оптимизатора для всех моделей использовался Adam [36], темп обучения варьировался от 10^{-5} до 10^{-3}, размер батча – от 32 до 128 (ниже сообщается лучшее качество для каждой модели). Гиперпараметры самих функций зафиксированы в соответствии с результатами авторов. При оптимизации использовался ранний останов, если качество не улучшалось в течение 10 эпох. В случае DistilBERT дополнительно использовался warmup на первых 100 итерациях. Для изображений при обучении используются стандартные методы аугментации: случайное отражение по горизонтали, вырезание и масштабирование случайного сегмента изображения. При замере качества тестовые изображения масштабируются до 256, а затем вырезается центральный фрагмент 224 × 224.

4.3. Результаты

Все описанные в разд. 3 функции потерь в данной работе сравниваются на рассмотренных выше задачах: основанных на распознавании изображений (Cars-196, CUB-200, SOP, Dogs-130) и естественного языка (News-20, WOS-134). Отметим, что многие методы по отдельности или небольшими наборами тестировалась на датасетах

Рис. 4. Dogs-130: случайная контрольная подвыборка после предобработки.

Таблица 2. Функционалы качества на контроле Cars-196

	R@1	R@2	R@4	R@8	R@16	R@32	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	86.14	91.80	95.15	97.15	98.32	99.17	47.23	33.05	90.49	70.59	74.69
Triplet Loss	85.68	91.80	95.61	97.42	98.41	99.11	49.21	35.05	90.34	73.39	77.10
Fast AP	83.36	90.01	93.95	96.38	97.79	98.78	48.92	34.41	88.49	70.47	74.60
Centroid Triplet Loss	83.80	90.82	95.08	7.63	98.90	**99.50**	40.99	27.04	89.17	70.38	74.48
Margin Loss	81.82	88.12	92.69	95.33	97.14	98.21	44.16	29.94	87.03	65.46	70.24
Multi Similarity Loss	86.89	92.58	95.62	97.58	98.74	99.19	50.99	36.74	91.12	73.84	77.49
SNN Loss	84.36	90.38	94.11	96.65	97.92	98.85	49.57	35.35	89.10	72.29	76.16
SupCon Loss	81.01	88.32	92.73	95.59	97.32	98.42	46.35	32.28	86.70	68.70	73.06
SNR Loss	86.88	92.24	95.27	97.31	98.61	99.28	48.41	34.00	91.00	70.11	74.27
Tuplet Margin Loss	**88.54**	**94.11**	**96.88**	**98.37**	**99.11**	99.49	50.53	36.24	**92.51**	**76.10**	**79.41**
Circle Loss	88.05	92.92	95.95	97.76	98.63	99.30	**52.81**	**38.46**	91.85	75.00	78.48
ArcFace	87.35	92.69	95.45	97.26	98.35	99.18	50.74	35.82	91.30	71.56	75.52
CosFace	87.06	92.32	95.23	97.21	98.34	99.08	50.94	35.72	91.06	72.84	76.62
SubCenter ArcFace	87.13	92.68	95.57	97.48	98.67	99.23	51.26	36.35	91.24	72.00	75.89
SoftTriple Loss	86.76	92.63	95.90	97.68	98.75	99.39	48.14	33.71	91.13	72.00	75.91
Proxy-Anchor Loss	88.43	93.48	96.06	97.86	98.88	99.43	52.30	36.34	92.16	74.42	77.92

Таблица 3. Функционалы качества на контроле 20 Newsgroups

	R@1	R@2	R@4	R@8	R@16	R@32	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	77.95	84.92	89.80	93.83	97.09	98.45	56.99	32.92	83.93	50.15	50.25
Triplet Loss	78.48	**85.41**	**90.44**	94.36	97.35	98.83	58.64	34.66	84.46	50.49	50.58
Fast AP	77.07	84.50	89.59	93.76	97.21	98.68	57.75	33.78	83.41	50.34	50.44
Centroid Triplet Loss	78.56	85.26	90.03	94.02	97.21	98.69	58.38	33.97	84.38	49.62	49.72
Margin Loss	77.16	84.72	89.72	93.85	97.25	98.54	58.04	34.03	83.52	49.46	49.56
Multi Similarity Loss	78.25	84.92	89.68	93.71	97.15	98.66	**59.01**	**35.40**	84.08	52.60	52.69
SNN Loss	77.70	84.37	89.48	93.82	97.08	98.54	58.34	35.00	83.67	50.78	50.88
SupCon Loss	77.44	84.56	89.68	93.92	97.30	98.67	57.92	34.19	83.63	50.52	50.61
SNR Loss	77.86	84.57	89.67	93.78	97.13	98.42	58.37	34.53	83.82	51.36	51.45
Tuplet Margin Loss	**78.65**	85.26	90.18	94.56	97.60	98.92	57.79	33.39	**84.51**	52.98	53.07
Circle Loss	78.53	85.23	90.34	**94.63**	**98.01**	**99.17**	56.84	32.24	84.50	**53.39**	**53.48**
ArcFace	76.24	83.95	89.05	93.87	97.25	98.79	56.56	33.32	82.84	49.97	50.06
CosFace	76.40	83.67	88.99	93.56	96.98	98.66	57.02	33.92	82.82	49.71	49.81
SubCenter ArcFace	76.51	83.79	89.09	93.27	96.76	98.31	57.61	34.85	82.82	49.97	50.07
SoftTriple Loss	77.02	83.86	89.26	93.61	97.13	98.75	57.22	33.86	83.18	48.95	49.05
Proxy-Anchor Loss	76.98	83.88	89.68	93.73	97.20	98.59	56.96	33.63	83.24	51.35	51.45

Cars-196, CUB-200 и SOP в статьях, где методы были впервые предложены – нередко авторские алгоритмы достигали наиболее высокого качества. Далее будут описаны результаты сравнения подходов нейросетевого обучения метрик в полностью равных условиях.

На Cars-196 табл. 2 с использованием ConvNeXT лучше всего сработал Tuplet Margin Loss, но самые высокие значения MAP-функционалов достигаются при использовании Circle Loss (9). Действительно, с точки зрения разных критериев качества разные методы могут оказываться лучше или хуже (нередко в статьях приводится только Recall@K). Отметим, что по MRR, NMI и AMI на Cars-196 также лучше сработал Tuplet Loss.

Рассмотрим News-20 (20 Newsgroups) – текстовый датасет с 20 классами и трансформерную модель: табл. 3 показывает, что на Recall лучше

Таблица 4. Функционалы качества на контроле CUB-200

	R@1	R@2	R@4	R@8	R@16	R@32	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	81.46	88.08	92.85	95.88	97.78	98.83	53.67	40.96	86.95	77.66	81.80
Triplet Loss	82.69	89.34	93.62	96.37	98.13	98.88	56.14	43.18	87.97	78.61	82.51
Fast AP	80.94	87.57	92.43	95.13	97.10	98.27	55.00	41.82	86.44	76.63	80.91
Centroid Triplet Loss	79.48	87.71	92.85	96.16	97.96	98.73	48.77	36.26	85.92	76.09	80.45
Margin Loss	77.36	85.13	90.53	94.72	96.83	98.05	49.53	36.50	83.84	70.00	75.46
Multi Similarity Loss	82.07	88.76	92.97	96.03	97.62	98.46	55.84	42.98	87.41	77.00	81.24
SNN Loss	82.01	88.68	93.21	96.06	97.67	98.57	56.64	43.58	87.39	77.97	82.02
SupCon Loss	80.19	87.52	92.34	95.67	97.39	98.42	55.03	41.82	86.10	76.43	80.77
SNR Loss	82.28	88.80	93.50	96.15	98.05	98.90	55.01	42.34	87.62	77.80	81.92
Tuplet Margin Loss	**83.36**	89.81	**93.96**	**96.89**	**98.18**	**99.02**	55.66	42.68	**88.54**	**79.88**	**83.55**
Circle Loss	83.35	**89.93**	93.74	96.28	97.93	98.83	**57.68**	**44.78**	88.44	79.29	83.13
ArcFace	80.61	87.22	91.78	94.89	97.06	98.40	54.70	41.44	86.11	73.91	78.79
CosFace	80.75	87.27	91.73	94.82	97.00	98.25	53.80	40.63	86.18	75.10	79.74
SubCenter ArcFace	80.12	87.15	92.04	94.87	96.77	98.00	54.72	41.57	85.83	73.87	78.69
SoftTriple Loss	80.82	87.74	92.43	95.76	97.66	98.69	53.48	40.50	86.49	77.07	81.32
Proxy-Anchor Loss	80.82	88.02	92.36	95.65	97.45	98.51	53.97	40.81	86.52	76.18	80.55

Таблица 5. Функционалы качества на контроле Dogs-130

	R@1	R@2	R@4	R@8	R@16	R@32	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	94.23	96.72	97.84	98.54	99.01	99.31	85.95	76.42	95.99	77.40	77.80
Triplet Loss	94.47	96.88	97.98	98.64	99.08	99.40	86.36	77.05	96.18	78.21	78.60
Fast AP	94.25	96.74	97.88	98.55	98.97	99.32	86.54	77.46	96.01	78.00	78.39
Centroid Triplet Loss	**94.57**	96.89	98.04	**98.74**	99.13	99.46	84.69	73.79	**96.25**	79.14	79.51
Margin Loss	94.30	96.78	97.86	98.59	99.00	99.29	86.47	77.36	96.05	77.98	78.38
Multi Similarity Loss	94.23	96.72	97.84	98.54	99.01	99.31	85.95	76.42	95.99	77.40	77.80
SNN Loss	94.28	96.68	97.89	98.55	98.97	99.28	**86.77**	**77.92**	96.02	77.89	78.29
SupCon Loss	94.27	96.75	97.90	98.58	99.00	99.36	86.18	76.84	96.04	77.66	78.06
SNR Loss	94.23	96.72	97.84	98.54	99.01	99.31	85.95	76.42	95.99	77.40	77.80
Tuplet Margin Loss	94.49	**96.92**	**98.09**	98.71	**99.16**	**99.46**	85.26	74.87	96.23	**79.43**	**79.81**
Circle Loss	94.36	96.84	97.88	98.55	98.97	99.31	86.40	77.27	96.09	77.80	78.20
ArcFace	94.29	96.69	97.83	98.51	98.99	99.32	85.90	76.38	96.01	77.75	78.15
CosFace	94.29	96.66	97.82	98.52	98.97	99.31	85.86	76.34	96.01	77.82	78.21
SubCenter ArcFace	94.27	96.75	97.87	98.52	99.02	99.32	85.99	76.48	96.02	77.47	77.87
SoftTriple Loss	94.25	96.67	97.80	98.53	98.99	99.31	86.06	76.80	95.99	77.96	78.36
Proxy-Anchor Loss	94.31	96.73	97.85	98.54	98.97	99.32	86.15	76.81	96.04	78.00	78.39

оптимизируют Tuplet Margin Loss и Triplet Loss, в то время как MAP-функционалы выше при использовании Multi Similarity Loss, кластерные – Circle Loss.

На CUB-200 (табл. 4) результаты во многом повторяют Cars-196: Tuplet Loss и Circle Loss имеют самое высокое качество.

Датасет Dogs-130 (табл. 5) достаточно простой для оптимизации: классы содержат особенно большое число примеров. Тем не менее на нем явно лучше оказалось использовать Tuplet Loss: как с точки зрения Recall, так и AMI / NMI. MAP же получился самым высоким у SNN Loss.

На SOP *(табл. 6: из рассмотренных в данном датасете самое большое число классов)* наиболее оптимален Circle Loss с точки зрения Recall и MAP-функционалов, хотя на AMI / NMI лучше себя показал Multi Similarity Loss.

Таблица 6. Функционалы качества на контроле SOP

	R@1	R@10	R@100	R@1000	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	81.08	90.60	95.91	98.70	63.56	54.87	84.45	54.14	90.87
Triplet Loss	80.31	91.61	96.92	99.20	61.69	53.02	84.30	50.82	90.16
Fast AP	80.80	91.69	96.89	99.17	63.51	54.94	84.63	54.19	90.83
Centroid Triplet Loss	76.55	89.22	95.83	98.88	56.33	47.22	81.01	45.90	89.11
Margin Loss	73.09	86.52	94.82	98.70	52.58	44.16	77.81	44.74	88.85
Multi Similarity Loss	81.32	91.30	96.56	98.99	64.05	55.71	84.83	**55.61**	**91.15**
SNN Loss	80.58	91.66	96.79	99.23	62.80	54.27	84.47	53.71	90.73
SupCon Loss	75.92	88.45	95.44	98.83	56.65	48.05	80.29	49.30	89.84
SNR Loss	81.98	91.55	96.38	98.82	64.98	56.28	85.36	54.24	90.91
Tuplet Margin Loss	50.72	66.17	79.99	92.71	34.63	22.88	56.11	24.10	84.29
Circle Loss	**82.14**	**92.50**	**97.08**	**99.25**	**65.27**	**56.87**	**85.80**	55.45	91.09
ArcFace	64.18	75.92	82.94	89.83	49.70	34.17	68.36	30.57	86.24
CosFace	64.17	75.64	82.94	89.91	49.27	33.89	68.21	30.10	86.17
SubCenter ArcFace	61.58	73.10	81.08	88.83	47.24	31.53	65.66	27.60	85.63
SoftTriple Loss	81.37	91.44	96.12	98.48	64.15	54.49	84.94	51.66	90.39
Proxy-Anchor Loss	80.68	91.48	96.33	98.60	63.55	53.63	84.52	51.68	90.35

Таблица 7. Функционалы качества на контроле WOS-134

	R@1	R@2	R@4	R@8	R@16	R@32	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	57.02	68.03	77.12	84.06	89.16	93.01	35.81	18.46	67.15	50.59	51.80
Triplet Loss	58.40	70.47	**79.88**	86.28	91.15	94.65	34.36	16.48	69.00	50.13	51.35
Fast AP	55.91	67.92	77.27	84.40	89.44	93.13	34.29	16.96	66.61	49.17	50.42
Centroid Triplet Loss	59.06	**70.76**	79.68	86.53	91.52	94.80	34.51	16.73	**69.40**	**51.81**	**52.99**
Margin Loss	57.22	68.74	78.02	85.22	90.15	93.79	33.10	15.43	67.65	48.61	49.88
Multi Similarity Loss	57.72	69.05	78.02	84.89	90.15	93.69	35.76	18.48	67.92	50.92	52.13
SNN Loss	57.96	69.44	78.19	85.09	90.10	93.72	35.56	17.96	68.16	50.19	51.41
SupCon Loss	57.91	69.76	79.02	85.85	90.69	94.25	34.63	16.99	68.43	50.23	51.45
SNR Loss	57.33	68.28	77.05	83.80	88.91	92.96	**35.87**	**18.67**	67.30	49.84	51.08
Tuplet Margin Loss	**59.08**	70.33	79.33	85.84	91.02	94.40	35.13	17.33	69.20	50.29	51.51
Circle Loss	58.50	70.10	79.61	**86.58**	**91.58**	**94.86**	34.06	16.22	68.97	51.43	52.62
ArcFace	54.00	66.36	76.20	83.90	89.47	93.60	30.61	13.59	65.17	45.45	46.80
CosFace	54.65	66.37	75.83	83.25	88.64	92.68	32.34	15.30	65.35	46.73	48.04
SubCenter ArcFace	55.85	67.40	76.90	84.00	89.34	93.12	33.65	16.49	66.40	48.95	50.21
SoftTriple Loss	55.90	67.48	76.70	84.05	89.30	93.05	32.29	15.07	66.40	47.21	48.51
Proxy-Anchor Loss	56.62	68.08	77.60	84.82	90.23	93.64	32.63	15.25	67.11	47.96	49.24

На заключительном текстовом датасете WOS-134 самые высокие MAP у SNR Loss (табл. 7), кластерные метрики же лучше у Centroid Triplet Loss. При этом на Recall@1 снова лучшим оказался Tuplet Margin Loss.

Полученные результаты для датасетов с изображениями можно рассматривать как новые бенчмарки для Deep Metric Learning – основанные не на Resnet или GoogleNet, а на ConvNeXT. По итогам экспериментов нередко методы имеют более высоке качество, чем в оригинальных статьях, например, на датасете SOP с ConvNeXT полученные в работе Recal@K выше, чем [9, 12, 13, 18, 22, 23]. Для использовавшихся в работе текстовых датасетов такая постановка задачи ранее рассматривалась в [37], но подход с применением трансформерной модели и функций потерь из

компьютерного зрения, вероятно, проверяется впервые.

5. ЗАКЛЮЧЕНИЕ

В работе сделан обзор функций потерь, которые ранее были предложены для оптимизации нейронных сетей в задачах глубокого обучения метрик. Были проведены эксперименты “в равных условиях” с описанными функциями потерь для разных доменов: изображения и тексты, а также с современными архитектурами нейронных сетей. Ожидаемо, выбор функции потерь зависит от задачи, модели и показателя качества. Тем не менее в большинстве проведенных экспериментов Tuplet Margin Loss и Circle Loss чаще других методов достигали наиболее хороших результатов, что немного удивительно, если учесть, что первый был предложен в 2019 г. и есть много “более свежих” методов.

Также отметим, что в экспериментах иногда удавалось добиться лучшего качества, чем в работах, в которых исследуемые функции потерь были предложены. Можно сделать достаточное число экспериментов при различных разбиениях датасетов на обучение и контроль с построением доверительных интервалов для полученных результатов, но ввиду трудоемкости такой серии экспериментов авторы ее не проводили. Из интересных направлений, в которых можно развить работу, следует упомянуть случай, когда метки принимают вещественные значения. В основном, функции потерь ориентированы на категориальные значения меток, поэтому в аналитической записи присутствуют объекты с равными и неравными метками. Интуитивно понятно, что в случае вещественных меток чем ближе метки, тем ближе должны быть представления, но полного исследования различных формализаций этой идеи пока не проводилось.

СПИСОК ЛИТЕРАТУРЫ

1. *Wei Chen, Yang Liu, Weiping Wang, Bakker E.M., Georgiou T.K.*, Paul Fieguth, Li Liu, Lew M.S.K. Deep image retrieval: A survey. ArXiv, 2021.
2. *Reimers N., Gurevych I.* Sentence-bert: Sentence embeddings using Siamese bert-networks. arXiv preprint arXiv:1908.10084, 2019.
3. *Iacopo Masi, Yue Wu, Tal Hassner, Prem Natarajan.* Deep face recognition: A survey. In 2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI). IEEE, 2018. P. 471–478.
4. *Mang Ye, Jianbing Shen, Gaojie Lin, Tao Xiang, Ling Shao, Steven CH Hoi.* Deep learning for person re-identification: A survey and outlook. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021.
5. *Musgrave K., Belongie S., Ser-Nam Lim.* A metric learning reality check. In European Conference on Computer Vision. Springer, 2020. P. 681–699.
6. *Johnson J., Douze M., Jégou H.* Billion-scale similarity search with GPUs. IEEE Transactions on Big Data. 2019. V. 7. № 3. P. 535–547.
7. *Chopra S., Hadsell R., LeCun Y.* Learning a similarity metric discriminatively, with application to face veri_-cation. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). IEEE. 2005. V. 1. P. 539–546.
8. *Schroff F., Kalenichenko D., Philbin J.* Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. P. 815–823.
9. *Fatih Cakir, Kun He, Xide Xia, Brian Kulis, Stan Sclaroff.* Deep metric learning to rank. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019. P. 1861–1870.
10. *Ustinova E., Lempitsky V.* Learning deep embeddings with histogram loss. Advances in Neural Information Processing Systems. 2016. P. 29.
11. *Wieczorek V., Rychalska B., Dąbrowski J.* On the unreasonable effectiveness of centroids in image retrieval. In International Conference on Neural Information Processing. Springer, 2021. P. 212–223.
12. *Chao-Yuan Wu, Manmatha R., Smola A.J., Krahenbuhl P.* Sampling matters in deep embedding learning. In Proceedings of the IEEE International Conference on Computer Vision. 2017. P. 2840–2848.
13. *Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, Scott M.R.* Multisimilarity loss with general pair weighting for deep metric learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019. P. 5022–5030.
14. *Frosst N., Papernot N., Hinton G.* Analyzing and improving representations with the soft nearest neighbor loss. In International conference on machine learning. PMLR, 2019. P. 2012–2020.
15. *Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, Dilip Krishnan*. Supervised contrastive learning. Advances in Neural Information Processing Systems. 2020. V. 33. P. 18661–18673.
16. *Tongtong Yuan, Weihong Deng, Jian Tang, Yinan Tang, Binghui Chen*. Signal-tonoise ratio: A robust distance metric for deep metric learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019. P. 4815–4824.
17. *Baosheng Yu, Dacheng Tao.* Deep metric learning with tuplet margin loss. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019. P. 6490–6499.
18. *Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, Yichen Wei.* Circle loss: A unified perspective of pair similarity optimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020. P. 6398–6407.
19. *Jiankang Deng, Jia Guo, Niannan Xue, Stefanos Zafeiriou.* Arcface: Additive angular margin loss for deep face recognition. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019. P. 4690–4699.

20. *Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, Wei Liu.* Cosface: Large margin cosine loss for deep face recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. P. 5265–5274.
21. *Jiankang Deng, Jia Guo, Tongliang Liu, Mingming Gong, Stefanos Zafeiriou.* Subcenter arcface: Boosting face recognition by large-scale noisy web faces. In European Conference on Computer Vision. Springer, 2020. P. 741–757.
22. *Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, Rong Jin.* Softtriple loss: Deep metric learning without triplet sampling. In Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019. P. 6450–6458.
23. *Sungyeon Kim, Dongwon Kim, Minsu Cho, Suha Kwak.* Proxy anchor loss for deep metric learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020. P. 3238–3247.
24. *Jonathan Krause, Michael Stark, Jia Deng, Li Fei-Fei.* 3d object representations for fine-grained categorization. In 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13), Sydney, Australia, 2013.
25. *Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, Serge Belongie.* The caltech-ucsd birds-200-2011 dataset. 2011.
26. *Hyun Oh Song, Yu Xiang, Stefanie Jegelka, Silvio Savarese.* Deep metric learning via lifted structured feature embedding. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
27. *Ding-Nan Zou, Song-Hai Zhang, Tai-Jiang Mu, Min Zhang.* A new dataset of dog breed images and a benchmark for finegrained classification. Computational Visual Media. 2020. V. 6. № 4. P. 477–487.
28. *Ken Lang.* Newsweeder: Learning to filter netnews. In Machine Learning Proceedings 1995, Elsevier, 1995. P. 331–339.
29. *Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, Laura E Barnes.* Hdltex: Hierarchical deep learning for text classification. In 2017 16th IEEE international conference on machine learning and applications (ICMLA). IEEE, 2017. P. 364–371.
30. *Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun.* Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. 770–778.
31. *Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich.* Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. P. 1–9.
32. *Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, Saining Xie.* A convnet for the 2020s. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.
33. *Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf.* Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108, 2019.
34. *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova.* Bert: Pretraining of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.
35. *Kevin Musgrave, Serge Belongie, Ser-Nam Lim.* Pytorch metric learning, 2020.
36. *Kingma D.P., Ba J.* Adam: A method for stochastic optimization. ArXiv preprint arXiv:1412.6980, 2014.
37. *Wohlwend J., Elenberg T.R., Altschul S., Henry S., Tao Lei.* Metric learning for dynamic text classification. arXiv preprint arXiv:1911.01026, 2019.

DEEP METRIC LEARNING: LOSS FUNCTIONS COMPARISON

R. L. Vasilev[a] and A. G. D'yakonov[b]

[a] *Yandex, Moscow, Russia*

[b] *Central University, Moscow, Russia*

Presented by Academician of the RAS A.A. Shananin

An overview of deep metric learning methods is presented. These methods have appeared in recent years, but were compared only with their predecessors, using neural networks of currently obsolete architectures to learn embeddings (on which the metric is calculated). The described methods were compared on different datasets from several domains, using pre-trained neural networks comparable in performance to SotA (state of the art): ConvNeXt for images, DistilBERT for texts. Labeled data sets were used, divided into two parts (train and test) in such a way that the classes did not overlap (i.e., for each class its objects are fully in train or fully in test). Such a large-scale honest comparison was made for the first time and led to unexpected conclusions: some "old" methods, for example, Tuplet Margin Loss, are superior in performance to their modern modifications and methods proposed in very recent works.

Keywords: machine learning, deep learning, metric, similarity