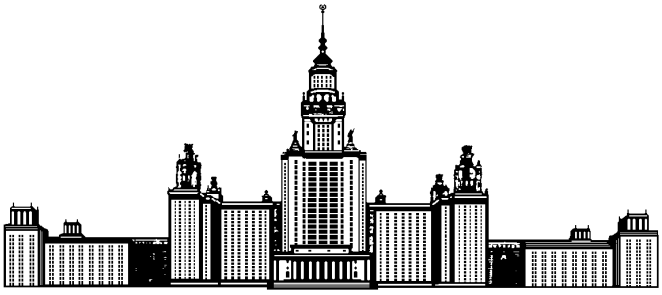

Московский государственный университет имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Васильев Руслан Леонидович

Нейросетевое обучение метрик

Deep Metric Learning

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
д.ф-м.н., профессор
Дьяконов Александр Геннадьевич

Москва, 2022

Содержание

1 Введение

Во множестве задач возникает потребность в оценке близости объектов. Поиск по изображениям [1] или текстовым документам [2], распознавание [3] и идентификация [4] лиц — современные методы решения данных задач используют *нейросетевое обучение метрик (Deep Metric Learning)*. «Обучить метрику» значит найти расстояние, которое между похожими объектами было бы мало, а между разными — велико.

В данной работе описываются и сравниваются современные нейросетевые методы обучения метрик, приводятся многочисленные эксперименты на разных моделях и задачах, причем и с изображениями, и с текстами.

В разд. 2 задача нейросетевого обучения метрик ставится формально.

В разд. 3 рассматриваются основные функционалы качества, позволяющие сравнить методы друг с другом.

В разд. 4 приводится обзор различных методов: основанных на сравнении представлений или на сведении к задаче классификации.

В разд. 5 описываются эксперименты с данными методами на разных датасетах, нейросетевых архитектурах и модальностях.

2 Постановка задачи

Рассмотрим множество объектов $x \in X$ и соответствующих им меток $y \in Y$. Объекты могут иметь произвольную природу (изображения, тексты, аудио, табличные данные и т.д). Общая задача *обучения метрик (Metric Learning)* — построить *метрику* $D_\theta(\cdot,\cdot) : X \times X \to \mathbb{R}$, удовлетворяющую следующим свойствам:

$$\begin{cases} \mathcal{D}_\theta(x_1, x_2) \to \min, & y_1 = y_2 \\ \mathcal{D}_\theta(x_1, x_2) \to \max, & y_1 \neq y_2 \end{cases} \tag{1}$$

То есть (1) формализует свойства, которые ожидаются от построенной метрики: *похожие объекты должны быть близки друг к другу и далеки от объектов другого класса*. Более того, объекты могут не иметь классов — нам в такой постановке достаточно лишь знания о том, является ли данная пара *позитивной* ($y_1 = y_2$) или *негативной* ($y_1 \neq y_2$).

В задаче *нейросетевого обучения метрик (Deep Metric Learning)* требуется обучить нейронную сеть — отображение $f_\theta(\cdot) : X \to \mathbb{R}^n$, то есть $\mathcal{D}_\theta$ параметризуется следующим образом:

$$\mathcal{D}_\theta(x_1, x_2) = \mathcal{D}(f_\theta(x_1), f_\theta(x_2)), \tag{2}$$

где в конечномерном пространстве R^n *векторных представлений (embeddings)* объектов используется метрика $\mathcal{D}$ — обычно евклидово расстояние.

Также часто ограничивают параметризацию единичной сферой: $\|f_\theta(x)\| = 1$ и вводят не $\mathcal{D}_\theta(x_1, x_2)$, а скалярное произведение:

$$\mathcal{S}_\theta(x_1, x_2) = \langle f_\theta(x_1), f_\theta(x_2) \rangle \tag{3}$$

Для поиска оптимальных параметров θ задается функция потерь $\mathcal{L}$ и оптимизируется на обучающей выборке. Различные варианты функционалов будут рассмотрены далее.

Случаи, когда задача обучения метрики предпочтительнее стандартных методов классификации:

- Число классов велико: $|Y| >> 1$;
- Не все $y \in Y$ представлены в обучающей выборке Y_{train};
- Отдельные классы содержат малое число объектов.

3 Критерии качества

3.1 Поисковые

Назовем *запросом (query)* объект, для которого производится поиск ближайших соседей, а объекты, среди которых производится поиск — *документами (documents)*.

Один из наиболее часто используемых критериев качества в *Metric Learning* — Recall@K, которая определяется как доля запросов, для которых среди K ближайших соседей нашелся релевантный документ *(отметим, что такое определение Recall@K используется именно в задачах обучения метрик — в информационном поиске под Recall@K обычно понимается доля найденных объектов среди релевантных)*.

Другой часто используемый функционал качества — MAP *(Mean Average Precision)*. Определим Precision@K как долю релевантных объектов среди K найденных. Если упорядочить для запроса индексы всех релевантных документов: $K_1, \ldots, K_R$, то можно вычислить

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

тогда MAP — 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

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

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

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

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

где rank_i — порядковый номер первого релевантного документа, n — общее число запросов.

3.2 Кластерные

Кроме поисковых критериев качества в *Metric Learning* нередко считается качество кластеризации в пространстве $f_\theta(X)$: ожидается, что в обученной модели похожие объекты будут образовывать группы. Недостаток данных критериев — зависимость функционала качества от алгоритма кластеризации.

Пусть $U = \{U_1, \ldots, U_n\}$ — истинное разбиение X, 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

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

где $P_U(i) = \frac{|U_i|}{|U|}$ — вероятность случайного объекта попасть в i-й кластер разбиения U, аналогично $P_V(i) = \frac{|V_i|}{|V|}$ и $P_{UV}(i, j) = \frac{|U_i \cap V_j|}{|V|}$ — вероятность 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:

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

Но чаще используют AMI:

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

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

4 Методы нейросетевого обучения метрик

4.1 Методы, основанные на сравнении представлений

4.1.1 Contrastive Loss

Contrastive Loss [7] — одна из первых функций потерь, предложенных для обучения метрик. Определяется следующим образом:

$$\mathcal{L} = \mathbb{1}\left\{y_i = y_j\right\} \left[\mathcal{D}_\theta\left(x_i, x_j\right) - m_p\right]_+^2 + \mathbb{1}\left\{y_i \neq y_j\right\} \left[m_n - \mathcal{D}_\theta\left(x_i, x_j\right)\right]_+^2 \tag{4}$$

То есть (4) 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.

4.1.2 Triplet Loss

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

$$\mathcal{L} = \left[\mathcal{D}_\theta(x_a, x_p)^2 - \mathcal{D}_\theta(x_a, x_n)^2 + m\right]_+ \tag{5}$$

Основное отличие (4) от (5): *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].

4.1.3 Fast AP

Как было замечено в разд. 3, 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. Авторы предлагают следующее приближение:

$$\text{FastAP} = \frac{1}{N_q^+} \sum_{j=1}^{L} \frac{H_j^+ h_j^+}{H_j}, \tag{6}$$

- 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 (6) используется *гистограммный биннинг (histogram binning)*, который в данном случае приближает истинное распределение *(а именно плотность и функцию распределения)* расстояний кусочно-постоянными функциями:
$$h(z) = \sum_{j=1}^{L} h_j \cdot \mathbb{1}\{z \in z_j\}, \quad H(z) = \sum_{j=1}^{L} H_j \cdot \mathbb{1}\{z \in z_j\}$$
- 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].

4.1.4 Centroid Triplet Loss

В [11] авторы предлагают модифицировать (5), заменив позитивные и негативные объекты (x_p, x_n) относительно x_a на центры их классов — (c_a, c_p):

$$\mathcal{L} = \left[\mathcal{D}_\theta(x_a, c_p)^2 - \mathcal{D}_\theta(x_a, c_n)^2 + m\right]_+ \tag{7}$$

Во время обучения центры классов считаются по батчу *(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).

4.1.5 Margin Loss

Авторы [12] предлагают использовать:

$$\mathcal{L} = \left[\alpha + \left(2 \cdot \mathbb{1}\{y_i = y_j\} - 1\right) \cdot \left(\mathcal{D}_\theta(x_i, x_j) - \beta\right)\right]_+ \tag{8}$$

Фактически (8) отличается от (4) 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.

4.1.6 Multi Similarity Loss

В [13] в функции потерь предлагается использовать больше информации о расстояниях между объектами в батче:

$$\mathcal{L} = \sum_{i=1}^{m} \left\{ \frac{1}{\alpha} \log\left[1 + \sum_{k \in \mathcal{P}_i} e^{-\alpha(S_{ik} - \lambda)}\right] + \frac{1}{\beta} \log\left[1 + \sum_{k \in \mathcal{N}_i} e^{\beta(S_{ik} - \lambda)}\right] \right\} \tag{9}$$

Здесь параметризуется $\mathcal{S}_\theta(x_i, x_j) = \langle f_\theta(x_i), f_\theta(x_j) \rangle$. Отметим, что выражение (9) можно рассматривать как гладкую аппроксимацию (4) (с дополнительными коэффициентами), включающую все попарные расстояния внутри батча.

4.1.7 SNN Loss

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

$$\mathcal{L} = -\frac{1}{m} \sum_{i=1}^{m} \log \frac{\sum_{j=1}^{m} \mathbb{1}\{y_i = y_j\} \exp\{\mathcal{S}_\theta(x_i, x_j)/\tau\}}{\sum_{j=1}^{m} \exp\{\mathcal{S}_\theta(x_i, x_j)/\tau\}} \tag{10}$$

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

4.1.8 SupCon Loss

В [15] авторы заметили, что в (10) позитивные примеры можно агрегировать по-разному, и предложили следующую альтернативу:

$$\mathcal{L} = -\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{m}\mathbb{1}\{y_i = y_j\}\log\frac{\exp\{\mathcal{S}_\theta(x_i, x_j)/\tau\}}{\sum_{j=1}^{m}\exp\{\mathcal{S}_\theta(x_i, x_j)/\tau\}} \tag{11}$$

Вариант (11) обычно оптимизируется лучше (10).

4.1.9 SNR Loss

В [16] используется (4), но вместо евклидовой метрики авторы оптимизируют SNR *(signal-to-noise ratio)*:

$$\mathcal{D}_\theta(x_i, x_j) = \frac{1}{\text{SNR}} = \frac{\text{Var}\left[f_\theta(x_i) - f_\theta(x_j)\right]}{\text{Var}\left[f_\theta(x_i)\right]} \tag{12}$$

Под Var в (12) понимается выборочная дисперсия. В отличие от евклидовой метрики, SNR-расстояние не симметрично, поэтому порядок элементов в паре имеет значение *(аналогично тому, как на Triplet Loss* (5) *влияет замена* $x_a \leftrightarrow x_p$*)*.

4.1.10 Tuplet Margin Loss

Авторы [17] внесли несколько существенных изменений в (5):

$$\mathcal{L} = \log\left(1 + \sum_{i=1}^{k-1} e^{s\left(\cos\theta_{a,n_i} - \cos\left(\theta_{a,p} - \beta\right)\right)}\right) \tag{13}$$

- В (13) 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 $\text{ReLU}(z) = [z]_+ = \max\{z, 0\}$ используется дифференцируемая $\text{Softplus}(z) = \log(1 + e^{s\cdot z})$, где s — коэффициент масштаба *(отвечает за радиус гиперсферы и влияет на скорость сходимости)*.

4.1.11 Circle Loss

В [18] предлагается оптимизировать модифицированную релаксацию (4):

$$\mathcal{L} = \log\left[1 + \sum_{j=1}^{L} e^{\gamma[s_n^j - m_n]_+ s_n^j}\sum_{i=1}^{K} e^{-\gamma[m_p - s_p^i]_+ s_p^i}\right] \tag{14}$$

Здесь s_p^j и s_n^i — функционалы сходства между позитивными и негативными парами.

4.2 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 подходы не гарантируют, что в итоговом пространстве объекты будут хорошо группироваться по классам.

4.2.1 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):

$$-\log \frac{e^{W_{y_i}^T f_\theta(x_i)}}{\sum_{j=1}^{c} e^{W_j^T f_\theta(x_i)}} \tag{15}$$

Рассмотрим j-й логит (15): $W_j^T f_\theta(x_i) = \left\|W_j\right\| \|f_\theta(x_i)\| \cos_{\theta_j}$, где θ_j — угол между W_j и $f_\theta(x_i)$. Зафиксируем норму всех весов $\left\|W_j\right\| = 1$, а также нормализуем пространство представлений: $\|f_\theta(x_i)\| = s$, в этом случае (15) примет вид (16):

$$\mathcal{L} = -\log \frac{e^{s\left(\cos(\theta_{y_i})\right)}}{e^{s\left(\cos(\theta_{y_i})\right)} + \sum_{j \neq y_i} e^{s\cos(\theta_j)}} \tag{16}$$

Добавив смещение к углу в (16), получим ArcFace Loss [19]:

$$\mathcal{L} = -\log \frac{e^{s\left(\cos(\theta_{y_i}+m)\right)}}{e^{s\left(\cos(\theta_{y_i}+m)\right)} + \sum_{j \neq y_i} e^{s\cos(\theta_j)}} \tag{17}$$

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

4.2.2 CosFace

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

$$\mathcal{L} = -\log \frac{e^{s\left(\cos(\theta_{y_i})+m\right)}}{e^{s\left(\cos(\theta_{y_i})+m\right)} + \sum_{j \neq y_i} e^{s\cos(\theta_j)}} \tag{18}$$

Хотя методы (17) и (18) нередко имеют практически одинаковое качество [19], в задаче распознавания лиц ArcFace часто работает чуть лучше.

4.2.3 SubCenter ArcFace

В [21] предлагается усилить устойчивость функционала (17) к шуму: для этого предлагается для каждого класса использовать K подцентров вместо одного. Вид функции потерь (17) прежний, но вводится K матриц весов $W^1, \ldots, W^K \in \mathbb{R}^{n \times c}$:

$$\theta_j = \arccos\left(\max_{k \in 1,\ldots,K} \left\{(W^k)_j^T f_\theta(x_i)\right\}\right), \tag{19}$$

то есть 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.

4.2.4 SoftTriple Loss

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

$$\mathcal{L} = -\log \frac{e^{s(G(i,y_i)+m)}}{e^{s(G(i,y_i)+m)} + \sum_{j \neq y_i} e^{sG(i,j)}}, \tag{20}$$

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

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

4.2.5 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 (17) центры классов θ_j можно рассматривать как прокси-объекты. В [23] используется следующая функция потерь:

$$\mathcal{L} = \frac{1}{|P^+|} \sum_{p \in P^+} \log\left(1 + \sum_{x \in X_p^+} e^{-\alpha(S(x,p)-m)}\right) + \frac{1}{|P|} \sum_{p \in P} \log\left(1 + \sum_{x \in X_p^-} e^{-\alpha(S(x,p)-m)}\right) \tag{21}$$

В (21) также есть гиперпараметры масштаба α и сдвига m, P — множество всех прокси-объектов, P^+ — положительных (в батче). X_p^-, X_p^+ и X_p^- — разбиение батча по прокси-объектам.

5 Эксперименты

5.1 Данные

Одной из важнейших особенностей *Deep Metric Learning* 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 тестовую выборки в соответствии с таблицей табл. 1.

5.1.1 Cars-196

Исходный датасет *Cars-196* [24] содержит 16 185 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*.

Рис. 1: Cars-196

5.1.2 CUB-200

В датасете *Caltech-UCSD Birds 200* [25] содержится 11 788 фотографий птиц, всего 200 категорий, примеры приведены на рис. 2

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

5.1.3 SOP

Датасет *Stanford Online Products* [26] содержит 120 053 изображений различных товаров, общее число категорий — 22 634, примеры приведены на рис. 3.

5.1.4 Dogs-130

В датасете *Tsinghua Dogs* [27] всего 70 432 различных фотографий собак, всего собрано 130 пород рис. 4.

Рис. 3: SOP: случайная тестовая подвыборка после предобработки

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

5.1.5 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 18 846 текстов и 20 классов.

5.1.6 WOS-134

В датасете *Web Of Science* [29] всего 46 985 текстовых документов. Число категорий достаточно велико: 134, поэтому на нем подходы *Deep Metric Learning* могут быть особенно актуальны.

	Число классов в обучении	*Число классов в тестовой выборке*	*Число объектов в обучении*	*Число объектов в тестовой 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

Таблица 1: Разбиение на обучающую и тестовую выборки, классы между разбиениями не пересекаются

5.2 Нейросетевые архитектуры

5.2.1 ConvNeXt

Большая часть рассмотренных ранее методов в оригинальных статьях сравнивалась на архитектурах 2014-2015 годов: *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.

5.2.2 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)*.

5.2.3 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.

5.3 Результаты

Все описанные в разд. 4 алгоритмы 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 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. Как показано в табл. 3, самые высокие значения MAP-функционалов достигаются при использовании Circle Loss (14). 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. 4 показывает, что на Recall лучше оптимизируют Tuplet

	R@1	R@2	R@4	R@8	R@16	R@32
Contrastive Loss	86.14	91.80	95.15	97.15	98.32	99.17
Triplet Loss	85.68	91.80	95.61	97.42	98.41	99.11
Fast AP	83.36	90.01	93.95	96.38	97.79	98.78
Centroid Triplet Loss	83.80	90.82	95.08	97.63	98.90	**99.50**
Margin Loss	81.82	88.12	92.69	95.33	97.14	98.21
Multi Similarity Loss	86.89	92.58	95.62	97.58	98.74	99.19
SNN Loss	84.36	90.38	94.11	96.65	97.92	98.85
SupCon Loss	81.01	88.32	92.73	95.59	97.32	98.42
SNR Loss	86.88	92.24	95.27	97.31	98.61	99.28
Tuplet Margin Loss	**88.54**	**94.11**	**96.88**	**98.37**	**99.11**	99.49
Circle Loss	88.05	92.92	95.95	97.76	98.63	99.30
ArcFace	87.35	92.69	95.45	97.26	98.35	99.18
CosFace	87.06	92.32	95.23	97.21	98.34	99.08
SubCenter ArcFace	87.13	92.68	95.57	97.48	98.67	99.23
SoftTriple Loss	86.76	92.63	95.90	97.68	98.75	99.39
Proxy-Anchor Loss	88.43	93.48	96.06	97.86	98.88	99.43

Таблица 2: Recall@K на датасете Cars-196 (тестовая выборка)

	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	47.23	33.05	90.49	70.59	74.69
Triplet Loss	49.21	35.05	90.34	73.39	77.10
Fast AP	48.92	34.41	88.49	70.47	74.60
Centroid Triplet Loss	40.99	27.04	89.17	70.38	74.48
Margin Loss	44.16	29.94	87.03	65.46	70.24
Multi Similarity Loss	50.99	36.74	91.12	73.84	77.49
SNN Loss	49.57	35.35	89.10	72.29	76.16
SupCon Loss	46.35	32.28	86.70	68.70	73.06
SNR Loss	48.41	34.00	91.00	70.11	74.27
Tuplet Margin Loss	50.53	36.24	**92.51**	**76.10**	**79.41**
Circle Loss	**52.81**	**38.46**	91.85	75.00	78.48
ArcFace	50.74	35.82	91.30	71.56	75.52
CosFace	50.94	35.72	91.06	72.84	76.62
SubCenter ArcFace	51.26	36.35	91.24	72.00	75.89
SoftTriple Loss	48.14	33.71	91.13	72.00	75.91
Proxy-Anchor Loss	52.30	36.34	92.16	74.42	77.92

Таблица 3: Поисковые и кластерные функционалы качества на датасете Cars-196 (тестовая выборка)

	R@1	R@2	R@4	R@8	R@16	R@32
Contrastive Loss	77.95	84.92	89.80	93.83	97.09	98.45
Triplet Loss	78.48	**85.41**	**90.44**	94.36	97.35	98.83
Fast AP	77.07	84.50	89.59	93.76	97.21	98.68
Centroid Triplet Loss	78.56	85.26	90.03	94.02	97.21	98.69
Margin Loss	77.16	84.72	89.72	93.85	97.25	98.54
Multi Similarity Loss	78.25	84.92	89.68	93.71	97.15	98.66
SNN Loss	77.70	84.37	89.48	93.82	97.08	98.54
SupCon Loss	77.44	84.56	89.68	93.92	97.30	98.67
SNR Loss	77.86	84.57	89.67	93.78	97.13	98.42
Tuplet Margin Loss	**78.65**	85.26	90.18	94.56	97.60	98.92
Circle Loss	78.53	85.23	90.34	**94.63**	**98.01**	**99.17**
ArcFace	76.24	83.95	89.05	93.87	97.25	98.79
CosFace	76.40	83.67	88.99	93.56	96.98	98.66
SubCenter ArcFace	76.51	83.79	89.09	93.27	96.76	98.31
SoftTriple Loss	77.02	83.86	89.26	93.61	97.13	98.75
Proxy-Anchor Loss	76.98	83.88	89.68	93.73	97.20	98.59

Таблица 4: Recall@K на датасете 20 Newsgroups (тестовая выборка)

	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	56.99	32.92	83.93	50.15	50.25
Triplet Loss	58.64	34.66	84.46	50.49	50.58
Fast AP	57.75	33.78	83.41	50.34	50.44
Centroid Triplet Loss	58.38	33.97	84.38	49.62	49.72
Margin Loss	58.04	34.03	83.52	49.46	49.56
Multi Similarity Loss	**59.01**	**35.40**	84.08	52.60	52.69
SNN Loss	58.34	35.00	83.67	50.78	50.88
SupCon Loss	57.92	34.19	83.63	50.52	50.61
SNR Loss	58.37	34.53	83.82	51.36	51.45
Tuplet Margin Loss	57.79	33.39	**84.51**	52.98	53.07
Circle Loss	56.84	32.24	84.50	**53.39**	**53.48**
ArcFace	56.56	33.32	82.84	49.97	50.06
CosFace	57.02	33.92	82.82	49.71	49.81
SubCenter ArcFace	57.61	34.85	82.82	49.97	50.07
SoftTriple Loss	57.22	33.86	83.18	48.95	49.05
Proxy-Anchor Loss	56.96	33.63	83.24	51.35	51.45

Таблица 5: Поисковые и кластерные функционалы качества на датасете 20 Newsgroups (тестовая выборка)

Margin Loss и Triplet Loss, в то время как по табл. 5 видим, что MAP-функционалы более высокие при использовании Multi Similarity Loss, а кластерные — Circle Loss.

На CUB-200 (табл. 6, табл. 10) результаты во многом повторяют Cars-196: Tuplet Loss и Circle Loss имеют самое высокое качество.

Датасет Dogs-130 (табл. 7, табл. 11) 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 *(из рассмотренных в данном датасете самое большое число классов)* наиболее оптимален Circle Loss с точки зрения Recall (табл. 8) и MAP-функционалов (табл. 12), хотя на AMI / NMI лучше себя показал Multi Similarity Loss.

На заключительном текстовом датасете WOS-134 самые высокие MAP у SNR Loss (табл. 13), кластерные метрики же лучше у Centroid Triplet Loss. При этом на Recall@1 (табл. 9) 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.

Итак, выбор оптимального метода действительно зависит от задачи, модели и критерия качества, тем не менее в большинстве проведенных экспериментов Tuplet Margin Loss и Circle Loss чаще других методов достигали наиболее хороших результатов.

6 Заключение

В данной работе были исследованы различные нейросетевые методы обучения метрики. Было показано, что хотя нет единственной лучшей функции потерь для всех задач, моделей и критериев качества, менее популярные методы тоже нередко оказываются лучшими. Также были получены новые бенчмарки для классических *(в задаче обучения метрик)* датасетов, часть из которых улучшила оригинальный результат. Предложено перенести подходы из распознавания изображений на текстовые задачи — произведены эксперименты и измерено качество.

На защиту выносится:

- Сравнение множества методов Deep Metric Learning: в экспериментах сопоставляются 11 constrastive-подходов и 5 классификационных, в равных условиях сравниваются методы для обработки как изображений, так и текстов — код доступен в репозитории работы[1];
- Получены новые бенчмарки, причем некоторые результаты превосходят качество оригинальных статей;
- Показано, что редко использующиеся методы *(по результатам экспериментов — [17, 18])* зачастую оказываются лучше более популярных подходов;
- Показано, что методы, традиционно использующиеся для идентификации лиц или поиска по изображениям, переносятся и на текстовые задачи.

[1] `https://github.com/artnitolog/deep_metric_learning`

Список литературы

1. *Chen W.*, *Liu Y.*, *Wang W.*, *Bakker E. M.*, *Georgiou T.*, *Fieguth P.*, *Liu L.*, *Lew M.* 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. *Masi I.*, *Wu Y.*, *Hassner T.*, *Natarajan P.* Deep face recognition: A survey // 2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI). — IEEE. 2018. — P. 471–478.
4. *Ye M.*, *Shen J.*, *Lin G.*, *Xiang T.*, *Shao L.*, *Hoi S. C.* Deep learning for person re-identification: A survey and outlook // IEEE Transactions on Pattern Analysis and Machine Intelligence. — 2021.
5. *Musgrave K.*, *Belongie S.*, *Lim S.-N.* A metric learning reality check // 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. — Vol. 7, no. 3. — P. 535–547.
7. *Chopra S.*, *Hadsell R.*, *LeCun Y.* Learning a similarity metric discriminatively, with application to face verification // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. — IEEE. 2005. — P. 539–546.
8. *Schroff F.*, *Kalenichenko D.*, *Philbin J.* Facenet: A unified embedding for face recognition and clustering // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2015. — P. 815–823.
9. *Cakir F.*, *He K.*, *Xia X.*, *Kulis B.*, *Sclaroff S.* Deep metric learning to rank // 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. — Vol. 29.
11. *Wieczorek M.*, *Rychalska B.*, *Dąbrowski J.* On the unreasonable effectiveness of centroids in image retrieval // International Conference on Neural Information Processing. — Springer. 2021. — P. 212–223.
12. *Wu C.-Y.*, *Manmatha R.*, *Smola A. J.*, *Krahenbuhl P.* Sampling matters in deep embedding learning // Proceedings of the IEEE International Conference on Computer Vision. — 2017. — P. 2840–2848.
13. *Wang X.*, *Han X.*, *Huang W.*, *Dong D.*, *Scott M. R.* Multi-similarity loss with general pair weighting for deep metric learning // 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 // International conference on machine learning. — PMLR. 2019. — P. 2012–2020.
15. *Khosla P.*, *Teterwak P.*, *Wang C.*, *Sarna A.*, *Tian Y.*, *Isola P.*, *Maschinot A.*, *Liu C.*, *Krishnan D.* Supervised contrastive learning // Advances in Neural Information Processing Systems. — 2020. — Vol. 33. — P. 18661–18673.

16. *Yuan T.*, *Deng W.*, *Tang J.*, *Tang Y.*, *Chen B.* Signal-to-noise ratio: A robust distance metric for deep metric learning // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2019. — P. 4815–4824.

17. *Yu B.*, *Tao D.* Deep metric learning with tuplet margin loss // Proceedings of the IEEE/CVF International Conference on Computer Vision. — 2019. — P. 6490–6499.

18. *Sun Y.*, *Cheng C.*, *Zhang Y.*, *Zhang C.*, *Zheng L.*, *Wang Z.*, *Wei Y.* Circle loss: A unified perspective of pair similarity optimization // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2020. — P. 6398–6407.

19. *Deng J.*, *Guo J.*, *Xue N.*, *Zafeiriou S.* Arcface: Additive angular margin loss for deep face recognition // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. — 2019. — P. 4690–4699.

20. *Wang H.*, *Wang Y.*, *Zhou Z.*, *Ji X.*, *Gong D.*, *Zhou J.*, *Li Z.*, *Liu W.* Cosface: Large margin cosine loss for deep face recognition // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2018. — P. 5265–5274.

21. *Deng J.*, *Guo J.*, *Liu T.*, *Gong M.*, *Zafeiriou S.* Sub-center arcface: Boosting face recognition by large-scale noisy web faces // European Conference on Computer Vision. — Springer. 2020. — P. 741–757.

22. *Qian Q.*, *Shang L.*, *Sun B.*, *Hu J.*, *Li H.*, *Jin R.* Softtriple loss: Deep metric learning without triplet sampling // Proceedings of the IEEE/CVF International Conference on Computer Vision. — 2019. — P. 6450–6458.

23. *Kim S.*, *Kim D.*, *Cho M.*, *Kwak S.* Proxy anchor loss for deep metric learning // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2020. — P. 3238–3247.

24. *Krause J.*, *Stark M.*, *Deng J.*, *Fei-Fei L.* 3D Object Representations for Fine-Grained Categorization // 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13). — Sydney, Australia, 2013.

25. *Wah C.*, *Branson S.*, *Welinder P.*, *Perona P.*, *Belongie S.* The caltech-ucsd birds-200-2011 dataset. — 2011.

26. *Song H. O.*, *Xiang Y.*, *Jegelka S.*, *Savarese S.* Deep Metric Learning via Lifted Structured Feature Embedding // IEEE Conference on Computer Vision and Pattern Recognition (CVPR). — 2016.

27. *Zou D.-N.*, *Zhang S.-H.*, *Mu T.-J.*, *Zhang M.* A new dataset of dog breed images and a benchmark for finegrained classification // Computational Visual Media. — 2020. — Vol. 6, no. 4. — P. 477–487.

28. *Lang K.* Newsweeder: Learning to filter netnews // Machine Learning Proceedings 1995. — Elsevier, 1995. — P. 331–339.

29. *Kowsari K.*, *Brown D. E.*, *Heidarysafa M.*, *Meimandi K. J.*, *Gerber M. S.*, *Barnes L. E.* Hdltex: Hierarchical deep learning for text classification // 2017 16th IEEE international conference on machine learning and applications (ICMLA). — IEEE. 2017. — P. 364–371.

30. *He K.*, *Zhang X.*, *Ren S.*, *Sun J.* Deep residual learning for image recognition // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — P. 770–778.

31. *Szegedy C.*, *Liu W.*, *Jia Y.*, *Sermanet P.*, *Reed S.*, *Anguelov D.*, *Erhan D.*, *Vanhoucke V.*, *Rabinovich A.* Going deeper with convolutions // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2015. — P. 1–9.

32. *Liu Z.*, *Mao H.*, *Wu C.-Y.*, *Feichtenhofer C.*, *Darrell T.*, *Xie S.* A ConvNet for the 2020s // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). — 2022.

33. *Sanh V.*, *Debut L.*, *Chaumond J.*, *Wolf T.* DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter // arXiv preprint arXiv:1910.01108. — 2019.

34. *Devlin J.*, *Chang M.-W.*, *Lee K.*, *Toutanova K.* Bert: Pre-training of deep bidirectional transformers for language understanding // arXiv preprint arXiv:1810.04805. — 2018.

35. *Musgrave K.*, *Belongie S.*, *Lim S.-N.* PyTorch Metric Learning. — 2020. — arXiv: 2008.09164 [cs.CV].

36. *Kingma D. P.*, *Ba J.* Adam: A method for stochastic optimization // arXiv preprint arXiv:1412.6980. — 2014.

37. *Wohlwend J.*, *Elenberg E. R.*, *Altschul S.*, *Henry S.*, *Lei T.* Metric learning for dynamic text classification // arXiv preprint arXiv:1911.01026. — 2019.

Приложения

A Recall@K

В данном приложении приводятся замеры на всех задачах, методах и моделях Recall@K (один из основных критериев качества в *Metric Learning*).

	R@1	R@2	R@4	R@8	R@16	R@32
Contrastive Loss	81.46	88.08	92.85	95.88	97.78	98.83
Triplet Loss	82.69	89.34	93.62	96.37	98.13	98.88
Fast AP	80.94	87.57	92.43	95.13	97.10	98.27
Centroid Triplet Loss	79.48	87.71	92.85	96.16	97.96	98.73
Margin Loss	77.36	85.13	90.53	94.72	96.83	98.05
Multi Similarity Loss	82.07	88.76	92.97	96.03	97.62	98.46
SNN Loss	82.01	88.68	93.21	96.06	97.67	98.57
SupCon Loss	80.19	87.52	92.34	95.67	97.39	98.42
SNR Loss	82.28	88.80	93.50	96.15	98.05	98.90
Tuplet Margin Loss	**83.36**	89.81	**93.96**	**96.89**	**98.18**	**99.02**
Circle Loss	83.35	**89.93**	93.74	96.28	97.93	98.83
ArcFace	80.61	87.22	91.78	94.89	97.06	98.40
CosFace	80.75	87.27	91.73	94.82	97.00	98.25
SubCenter ArcFace	80.12	87.15	92.04	94.87	96.77	98.00
SoftTriple Loss	80.82	87.74	92.43	95.76	97.66	98.69
Proxy-Anchor Loss	80.82	88.02	92.36	95.65	97.45	98.51

Таблица 6: Recall@K на датасете CUB-200 (тестовая выборка)

	R@1	R@2	R@4	R@8	R@16	R@32
Contrastive Loss	94.23	96.72	97.84	98.54	99.01	99.31
Triplet Loss	94.47	96.88	97.98	98.64	99.08	99.40
Fast AP	94.25	96.74	97.88	98.55	98.97	99.32
Centroid Triplet Loss	**94.57**	96.89	98.04	**98.74**	99.13	99.46
Margin Loss	94.30	96.78	97.86	98.59	99.00	99.29
Multi Similarity Loss	94.23	96.72	97.84	98.54	99.01	99.31
SNN Loss	94.28	96.68	97.89	98.55	98.97	99.28
SupCon Loss	94.27	96.75	97.90	98.58	99.00	99.36
SNR Loss	94.23	96.72	97.84	98.54	99.01	99.31
Tuplet Margin Loss	94.49	**96.92**	**98.09**	98.71	**99.16**	**99.46**
Circle Loss	94.36	96.84	97.88	98.55	98.97	99.31
ArcFace	94.29	96.69	97.83	98.51	98.99	99.32
CosFace	94.29	96.66	97.82	98.52	98.97	99.31
SubCenter ArcFace	94.27	96.75	97.87	98.52	99.02	99.32
SoftTriple Loss	94.25	96.67	97.80	98.53	98.99	99.31
Proxy-Anchor Loss	94.31	96.73	97.85	98.54	98.97	99.32

Таблица 7: Recall@K на датасете Dogs-130 (тестовая выборка)

	R@1	R@10	R@100	R@1000
Contrastive Loss	81.08	90.60	95.91	98.70
Triplet Loss	80.31	91.61	96.92	99.20
Fast AP	80.80	91.69	96.89	99.17
Centroid Triplet Loss	76.55	89.22	95.83	98.88
Margin Loss	73.09	86.52	94.82	98.70
Multi Similarity Loss	81.32	91.30	96.56	98.99
SNN Loss	80.58	91.66	96.79	99.23
SupCon Loss	75.92	88.45	95.44	98.83
SNR Loss	81.98	91.55	96.38	98.82
Tuplet Margin Loss	50.72	66.17	79.99	92.71
Circle Loss	**82.14**	**92.50**	**97.08**	**99.25**
ArcFace	64.18	75.92	82.94	89.83
CosFace	64.17	75.64	82.94	89.91
SubCenter ArcFace	61.58	73.10	81.08	88.83
SoftTriple Loss	81.37	91.44	96.12	98.48
Proxy-Anchor Loss	80.68	91.48	96.33	98.60

Таблица 8: Recall@K на датасете SOP (тестовая выборка)

	R@1	R@2	R@4	R@8	R@16	R@32
Contrastive Loss	57.02	68.03	77.12	84.06	89.16	93.01
Triplet Loss	58.40	70.47	**79.88**	86.28	91.15	94.65
Fast AP	55.91	67.92	77.27	84.40	89.44	93.13
Centroid Triplet Loss	59.06	**70.76**	79.68	86.53	91.52	94.80
Margin Loss	57.22	68.74	78.02	85.22	90.15	93.79
Multi Similarity Loss	57.72	69.05	78.02	84.89	90.15	93.69
SNN Loss	57.96	69.44	78.19	85.09	90.10	93.72
SupCon Loss	57.91	69.76	79.02	85.85	90.69	94.25
SNR Loss	57.33	68.28	77.05	83.80	88.91	92.96
Tuplet Margin Loss	**59.08**	70.33	79.33	85.84	91.02	94.40
Circle Loss	58.50	70.10	79.61	**86.58**	**91.58**	**94.86**
ArcFace	54.00	66.36	76.20	83.90	89.47	93.60
CosFace	54.65	66.37	75.83	83.25	88.64	92.68
SubCenter ArcFace	55.85	67.40	76.90	84.00	89.34	93.12
SoftTriple Loss	55.90	67.48	76.70	84.05	89.30	93.05
Proxy-Anchor Loss	56.62	68.08	77.60	84.82	90.23	93.64

Таблица 9: Recall@K на датасете WOS-134 (тестовая выборка)

B Другие функционалы качества

В данном приложении приводятся поисковые и кластерные критерии качества разд. 3, измеренные на всех рассмотренных в работе моделях и задачах.

	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	53.67	40.96	86.95	77.66	81.80
Triplet Loss	56.14	43.18	87.97	78.61	82.51
Fast AP	55.00	41.82	86.44	76.63	80.91
Centroid Triplet Loss	48.77	36.26	85.92	76.09	80.45
Margin Loss	49.53	36.50	83.84	70.00	75.46
Multi Similarity Loss	55.84	42.98	87.41	77.00	81.24
SNN Loss	56.64	43.58	87.39	77.97	82.02
SupCon Loss	55.03	41.82	86.10	76.43	80.77
SNR Loss	55.01	42.34	87.62	77.80	81.92
Tuplet Margin Loss	55.66	42.68	**88.54**	**79.88**	**83.55**
Circle Loss	**57.68**	**44.78**	88.44	79.29	83.13
ArcFace	54.70	41.44	86.11	73.91	78.79
CosFace	53.80	40.63	86.18	75.10	79.74
SubCenter ArcFace	54.72	41.57	85.83	73.87	78.69
SoftTriple Loss	53.48	40.50	86.49	77.07	81.32
Proxy-Anchor Loss	53.97	40.81	86.52	76.18	80.55

Таблица 10: Поисковые и кластерные функционалы качества на датасете CUB-200 (тестовая выборка)

	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	85.95	76.42	95.99	77.40	77.80
Triplet Loss	86.36	77.05	96.18	78.21	78.60
Fast AP	86.54	77.46	96.01	78.00	78.39
Centroid Triplet Loss	84.69	73.79	**96.25**	79.14	79.51
Margin Loss	86.47	77.36	96.05	77.98	78.38
Multi Similarity Loss	85.95	76.42	95.99	77.40	77.80
SNN Loss	**86.77**	**77.92**	96.02	77.89	78.29
SupCon Loss	86.18	76.84	96.04	77.66	78.06
SNR Loss	85.95	76.42	95.99	77.40	77.80
Tuplet Margin Loss	85.26	74.87	96.23	**79.43**	**79.81**
Circle Loss	86.40	77.27	96.09	77.80	78.20
ArcFace	85.90	76.38	96.01	77.75	78.15
CosFace	85.86	76.34	96.01	77.82	78.21
SubCenter ArcFace	85.99	76.48	96.02	77.47	77.87
SoftTriple Loss	86.06	76.80	95.99	77.96	78.36
Proxy-Anchor Loss	86.15	76.81	96.04	78.00	78.39

Таблица 11: Поисковые и кластерные функционалы качества на датасете Dogs-130 (тестовая выборка)

	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	63.56	54.87	84.45	54.14	90.87
Triplet Loss	61.69	53.02	84.30	50.82	90.16
Fast AP	63.51	54.94	84.63	54.19	90.83
Centroid Triplet Loss	56.33	47.22	81.01	45.90	89.11
Margin Loss	52.58	44.16	77.81	44.74	88.85
Multi Similarity Loss	64.05	55.71	84.83	**55.61**	**91.15**
SNN Loss	62.80	54.27	84.47	53.71	90.73
SupCon Loss	56.65	48.05	80.29	49.30	89.84
SNR Loss	64.98	56.28	85.36	54.24	90.91
Tuplet Margin Loss	34.63	22.88	56.11	24.10	84.29
Circle Loss	**65.27**	**56.87**	**85.80**	55.45	91.09
ArcFace	49.70	34.17	68.36	30.57	86.24
CosFace	49.27	33.89	68.21	30.10	86.17
SubCenter ArcFace	47.24	31.53	65.66	27.60	85.63
SoftTriple Loss	64.15	54.49	84.94	51.66	90.39
Proxy-Anchor Loss	63.55	53.63	84.52	51.68	90.35

Таблица 12: Поисковые и кластерные функционалы качества на датасете SOP (тестовая выборка)

	MAP	MAP@R	MRR	AMI	NMI
Contrastive Loss	35.81	18.46	67.15	50.59	51.80
Triplet Loss	34.36	16.48	69.00	50.13	51.35
Fast AP	34.29	16.96	66.61	49.17	50.42
Centroid Triplet Loss	34.51	16.73	**69.40**	**51.81**	**52.99**
Margin Loss	33.10	15.43	67.65	48.61	49.88
Multi Similarity Loss	35.76	18.48	67.92	50.92	52.13
SNN Loss	35.56	17.96	68.16	50.19	51.41
SupCon Loss	34.63	16.99	68.43	50.23	51.45
SNR Loss	**35.87**	**18.67**	67.30	49.84	51.08
Tuplet Margin Loss	35.13	17.33	69.20	50.29	51.51
Circle Loss	34.06	16.22	68.97	51.43	52.62
ArcFace	30.61	13.59	65.17	45.45	46.80
CosFace	32.34	15.30	65.35	46.73	48.04
SubCenter ArcFace	33.65	16.49	66.40	48.95	50.21
SoftTriple Loss	32.29	15.07	66.40	47.21	48.51
Proxy-Anchor Loss	32.63	15.25	67.11	47.96	49.24

Таблица 13: Поисковые и кластерные функционалы качества на датасете WOS-134 (тестовая выборка)