Московский государственный университет имени М.В. Ломоносова
Механико-математический факультет
Кафедра Математической теории интеллектуальных систем (МаТИС)

Курсовая работа

CS-Loss - кластерный подход в обучении сиамских сетей

Макаров Илья Александрович
Научный руководитель:
Миронов Андрей Михайлович

Москва, 2025

Содержание

Введение

Современные технологии машинного обучения, в частности нейронные сети, активно применяются в задачах компьютерного зрения, включая распознавание и верификацию лиц. Одним из ключевых инструментов в этой области являются сиамские нейронные сети, которые позволяют сравнивать изображения, определяя степень их схожести. Однако их эффективность во многом зависит от выбранной функции потерь, которая определяет, как модель обучается различать объекты.

Традиционные подходы, такие как Triplet Loss, сталкиваются с рядом проблем: необходимость сложных стратегий майнинга триплетов, чувствительность к гиперпараметрам и риск нестабильности обучения. В то же время более современные методы, такие как ArcFace или CosFace, хоть и демонстрируют высокую точность, теряют геометрическую интерпретируемость, что усложняет их анализ и адаптацию.

В данной работе предлагается новый подход к обучению сиамских сетей — CS-Loss (Cluster Separation Loss), который сочетает простоту вычислений, ясную геометрическую интерпретацию и эффективность. Основная идея заключается в одновременной минимизации расстояний между объектами одного класса (компактность кластеров) и максимизации расстояний между центрами разных классов (разделение кластеров).

Актуальность

Актуальность исследования функций потерь в сиамских нейронных сетях для верификации лиц обусловлена возрастающей ролью биометрической аутентификации в современных системах безопасности, персонализированных сервисах и автоматизированном контроле доступа. В условиях роста требований к надёжности идентификации ключевое значение приобретает эффективность алгоритмов распознавания, которая во многом зависит от выбора и оптимизации функций потерь. От нее зависят как качество полученной модели, так и скорость, и сложность обучения.

Практическая значимость

Практическая значимость работы заключается в том, что её результаты могут быть использованы для улучшения реальных биометрических систем. Введенная в данной работе CS-Loss (Cluster Separation Loss) показала очень хорошие результаты в сравнении с другой популярной функцией потерь triplet loss. Помимо хороших показателей, она остается интерпретируемой, что также является немаловажным фактором. Также в данной работе проведено сравнение методов майнинга триплетов, что также может помочь при реализации архитектур, обучаемых при помощи triplet loss или похожих функций потерь.

Литературный обзор

Сиамские нейронные сети

Сиамские сети состоят из двух или более идентичных подсетей с общими весами, которые обрабатывают несколько входных изображений для создания их векторных представлений (эмбеддингов). Эта архитектура была первоначально разработана для верификации подписей и позже адаптирована для задач верификации лиц. Ключевая идея заключается в создании пространства признаков, где изображения одного человека отображаются близко друг к другу, а изображения разных людей — далеко друг от друга.

Данная архитектура находит применение при решении таких задач как:

- Задачи верификации (лиц, подписей) [1]
- Поиск дубликатов [2]
- One-shot learning [3]
- Ранжирование в рекомендательных системах [4]
- Поиска похожих объектов [5]

Сверточные нейронные сети

При решении задач, нацеленных на обработку изображений и извлечения из них какой-либо информации, часто применяются сверточные нейронные сети (CNN). Одним из первых применений CNN стало решение задачи классификации изображений [6]. Их простота и эффективность стали ключевыми факторами для применения их почти во всех задачах, связанных с обработкой изображений. Именно они применяются при решении задачи верификации лиц, например в работах [7] и [1].

Функции потерь для задачи верификации лиц

Одним из основных факторов успеха при обучении нейронных сетей является функция потерь. Для рассматриваемой задачи существует большое их число. Одной из первых предложенных стала contrastive loss [8].

Для пары входных векторов (x_i, x_j) контрастная функция потерь определяется как:

$$\mathcal{L}(x_i, x_j, y_{ij}) = \begin{cases} \frac{1}{2}\|f(x_i) - f(x_j)\|_2^2 & \text{если } y_{ij} = 1 \text{ (схожие объекты)} \\ \frac{1}{2}\max(0, \alpha - \|f(x_i) - f(x_j)\|_2)^2 & \text{если } y_{ij} = 0 \text{ (различные объекты)} \end{cases}$$

- $f(x)$ – выходной эмбеддинг нейронной сети
- y_{ij} – бинарная метка (1 если объекты схожи, 0 если различны)
- $\alpha > 0$ – гиперпараметр **margin**, определяющий минимально желаемое расстояние между разными объектами

Позже предложили triplet loss [1], в нем обучение происходит на тройках изображений, называемых триплетами.

$$\mathcal{L}_{\text{triplet}} = \max\left(\|f(A) - f(P)\|_2^2 - \|f(A) - f(N)\|_2^2 + \alpha, 0\right)$$

- $\mathcal{L}_{\text{triplet}}$ - значение функции потерь
- A (**Anchor**) - опорный пример
- P (**Positive**) - позитивный пример (того же класса)
- N (**Negative**) - негативный пример (другого класса)
- $f(\cdot)$ - функция эмбеддинга (например, нейросеть)
- α - параметр отступа (**margin**)

Данная функция потерь имеет простую геометрическую интерпретацию, однако она учитывает лишь малую долю информации, заключенной в триплетах. Конечно, нам хотелось бы использовать при обучении информацию, заключенную во всех изображениях класса, которому принадлежит **Anchor**. Помимо этого, нам необходимо некоторое правило, по которому мы будет составлять триплеты. Наиболее широко используются варианты:

- **Сложные триплеты** (Hard triplets) - предложены в [9]:
  - В качестве позитивного примера берут наиболее удаленного от якоря представителя данного класса, в качестве негативного примера берут наиболее близкого представителя другого класса
  - Наиболее информативны для обучения, но могут содержать шум
  - Сложны для оптимизации, могут привести к коллапсу модели
- **Полусложные триплеты** (Semi-hard triplets):
  - Позитивный пример берется наиболее удаленным, а негативный исходя из соотношения: $d(A, N) > d(A, P)$ но $d(A, P) + \alpha > d(A, N)$
  - Оптимальный баланс между информативностью и стабильностью обучения
  - Помогают избежать проблем сходимости
- **Случайные триплеты** (Random triplets):
  - Полностью случайные комбинации якорей, позитивов и негативов
  - Просты в генерации, но содержат много неинформативных примеров
  - Используются на начальных этапах обучения

Позже были предложены SphereFace [10], CosFace [11] и ArcFace [7], оказавшиеся эффективнее triplet loss, но они теряют геометрическую интерпретацию, так как обучение происходит на задаче классификации полученных эмбеддингов.

CS-Loss (Cluster Separation Loss)

Созданная в рамках данной работы функция потерь решает обозначенные выше недостатки других функций потерь. Она состоит из двух компонент:

$$\mathcal{L}_{\text{CS-Loss}} = \alpha \cdot \mathcal{L}_{\text{compactness}} + \mathcal{L}_{\text{separation}}$$

Компонента компактности кластеров

$$\mathcal{L}_{\text{compactness}} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{N_k} \sum_{i=1}^{N_k} \text{ReLU}\left(\|\mathbf{c}_k - \mathbf{x}_i^k\|_2 - \delta_{\text{close}}\right)$$

- K – количество кластеров
- N_k – количество элементов в кластере k
- $\mathbf{c}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \mathbf{x}_i^k$ – центр кластера k
- $\mathbf{x}_i^k$ – i-й элемент кластера k
- $\delta_{\text{close}} = 0.1$ – пороговое расстояние
- $\alpha = 0.4$ – гиперпараметр

Компонента разделения кластеров

$$\mathcal{L}_{\text{separation}} = \frac{1}{K} \sum_{k=1}^{K} \text{ReLU}\left(\delta_{\text{far}} - \|\mathbf{c}_k - \mathbf{c}_{\text{nearest}(k)}\|_2\right)$$

- $\mathbf{c}_{\text{nearest}(k)}$ – центр ближайшего кластера к кластеру k
- $\delta_{\text{far}} = 0.5$ – минимальное желаемое расстояние между центрами

Замечания

- У функции потерь простая геометрическая интерпретации: она стремится сблизить вектора одного кластера и отдалить кластеры друг от друга.
- Функция использует всю информацию, заключенную в кластере.
- Функция проста в вычислении.

Формальная постановка задачи верификации лиц

Исходные данные

Пусть заданы:

- Множество изображений $\mathcal{X} = \{x_1, \ldots, x_N\}$
- Соответствующие идентификаторы классов $\mathcal{Y} = \{y_1, \ldots, y_M\}$, где $y_i \in \{1, \ldots, M\}$
- Функция эмбеддинга $f : \mathcal{X} \to R^d$, преобразующая изображение в вектор признаков
- Функция расстояния $\rho : R^d \times R^d \to R^+$

Постановка задачи

Для произвольной пары изображений $(x_i, x_j) \in \mathcal{X} \times \mathcal{X}$ требуется определить находятся ли на них один и тот же объект или нет, для этого используется следующая последовательность действий:

1. Получить эмбеддинги изображений
2. Посчитать расстояние между ними
3. В зависимости от порога сделать вывод о том, один и тот же объект на фотографии или нет

Подготовка экспериментальной части

Датасет

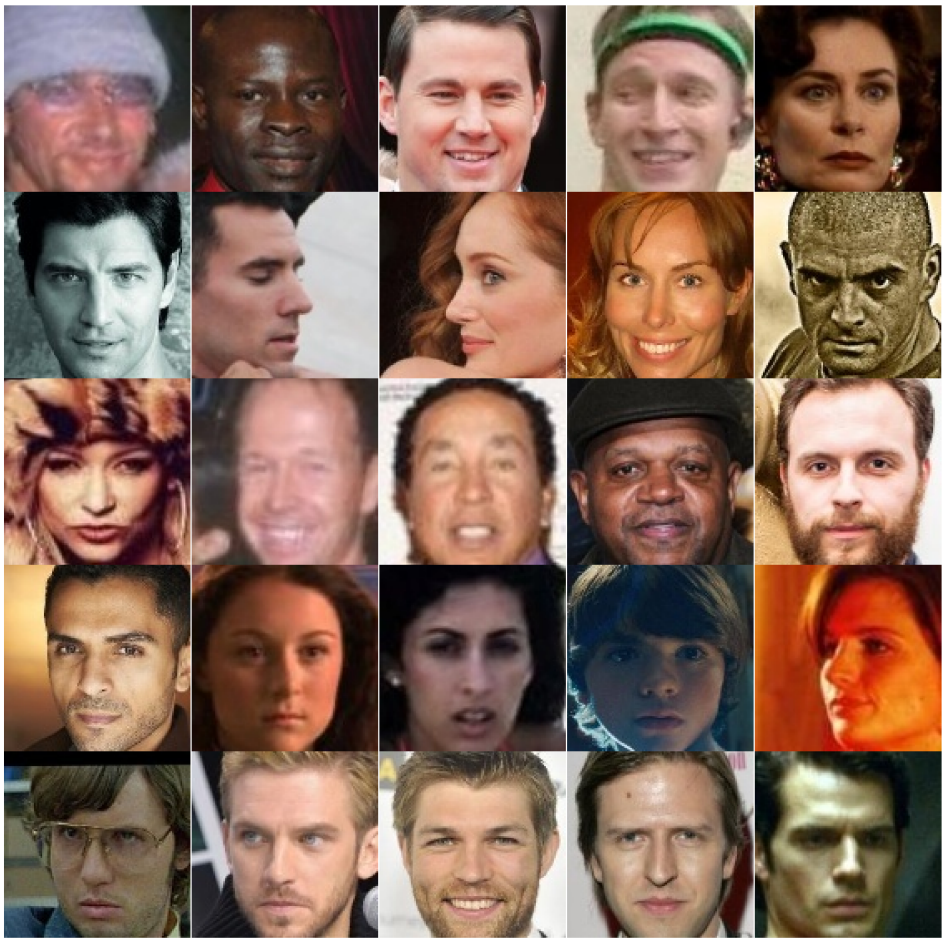

Рис. 1: Пример данных из датасета

Используется датасет `CASIA-WEBFACE`, содержащий фотографии людей размером 112×112 пикселей. В нем представлено 10 572 личности, при этом на одну личность приходится от 9 до более чем 1000 фотографий.

Обработка датасета

- Личности, на которых приходится менее 10 фотографий, были исключены из датасета.
- Личности с количеством фотографий от 10 до 16 включены в тестовую выборку.
- Остальные личности вошли в тренировочную выборку.

- Была проведена аугментация изображений: случайные повороты на угол, имеющий нормальное распределение со стандартным отклонением в $\frac{3\pi}{50}$.

Экспериментальная часть

Метрики качества

Точность (Accuracy)

Точность – доля правильно классифицированных примеров среди всех предсказаний:

$$\text{Accuracy} = \frac{\text{Число верно классифицированных пар}}{\text{Общее число пар}}$$

VAL@FAR(10^{-2})

Метрика Verification Accuracy at Fixed False Acceptance Rate (10^{-2}) показывает точность верификации при фиксированном уровне ложных принятий 1%:

$$\text{VAL}(d) = \frac{|\text{TA}(d)|}{|\mathcal{P}_{\text{same}}|}$$

$$\text{FAR}(d) = \frac{|\text{FA}(d)|}{|\mathcal{P}_{\text{diff}}|}$$

- $\mathcal{P}_{\text{same}}$ – множество всех пар изображений одного человека (genuine pairs)
- $\mathcal{P}_{\text{diff}}$ – множество всех пар изображений разных людей (impostor pairs)
- $\text{TA}(d) = \{(x_i, x_j) \in \mathcal{P}_{\text{same}} \mid \text{dist}(x_i, x_j) \leq d\}$ – множество верно принятых пар
- $\text{FA}(d) = \{(x_i, x_j) \in \mathcal{P}_{\text{diff}} \mid \text{dist}(x_i, x_j) \leq d\}$ – множество ложных принятий

Процедура вычисления:

1. Для всех пар вычисляются расстояния между эмбеддингами
2. Подбирается порог t, при котором $\text{FAR} = 0.01$
3. Вычисляется VAL при найденном пороге t

Формирование выборки для стохастического градиентного спуска

- Для **cluster loss**:
  - Из тренировочной выборки извлекаются 96 личностей с вероятностями, пропорциональными числу изображений на каждую из них.
  - Для каждой личности равновероятно извлекаются 15 изображений.
- Для **triplet loss**:
  - Аналогично *CS-Loss* извлекаются 192 личности по 15 изображений на каждую.
  - Для каждой личности выбираются 5 якорей (A).
  - По оставшейся выборке формируются 5 триплетов согласно способу майнинга (большее кол-во приводит к переизбытку одинаковых или похожих триплетов и малой информативности каждого из них).
  - В итоге получается 960 триплетов.

Валидация

- Формируются пары в равной пропорции: фотографии одного и того же класса и фотографии разных классов (чтобы избежать дисбаланса классов).
- Вычисляются оптимальный порог (*threshold*) и значения метрик.

Архитектура

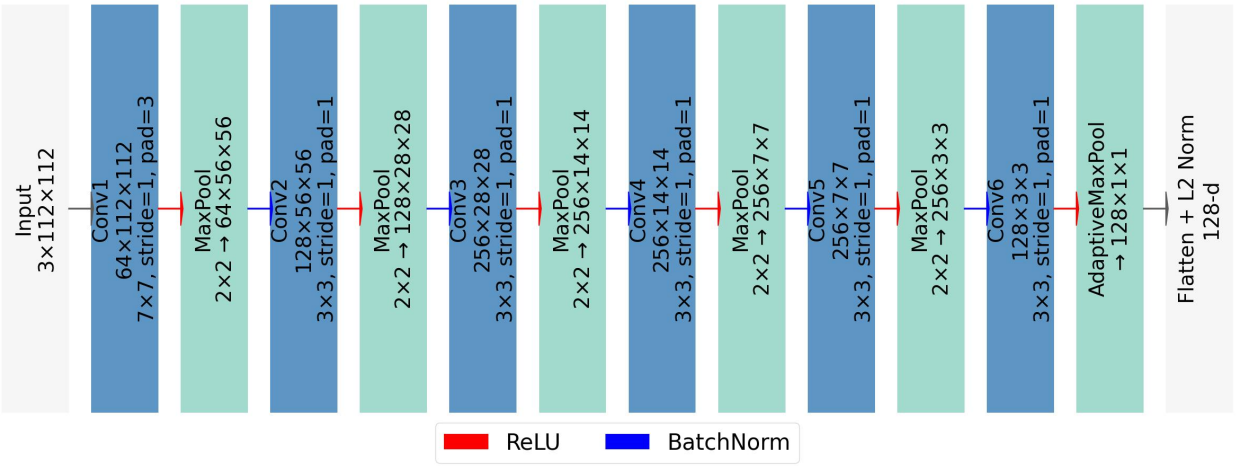

Рис. 2: Архитектура сети

Результаты

Обучение с CS-Loss

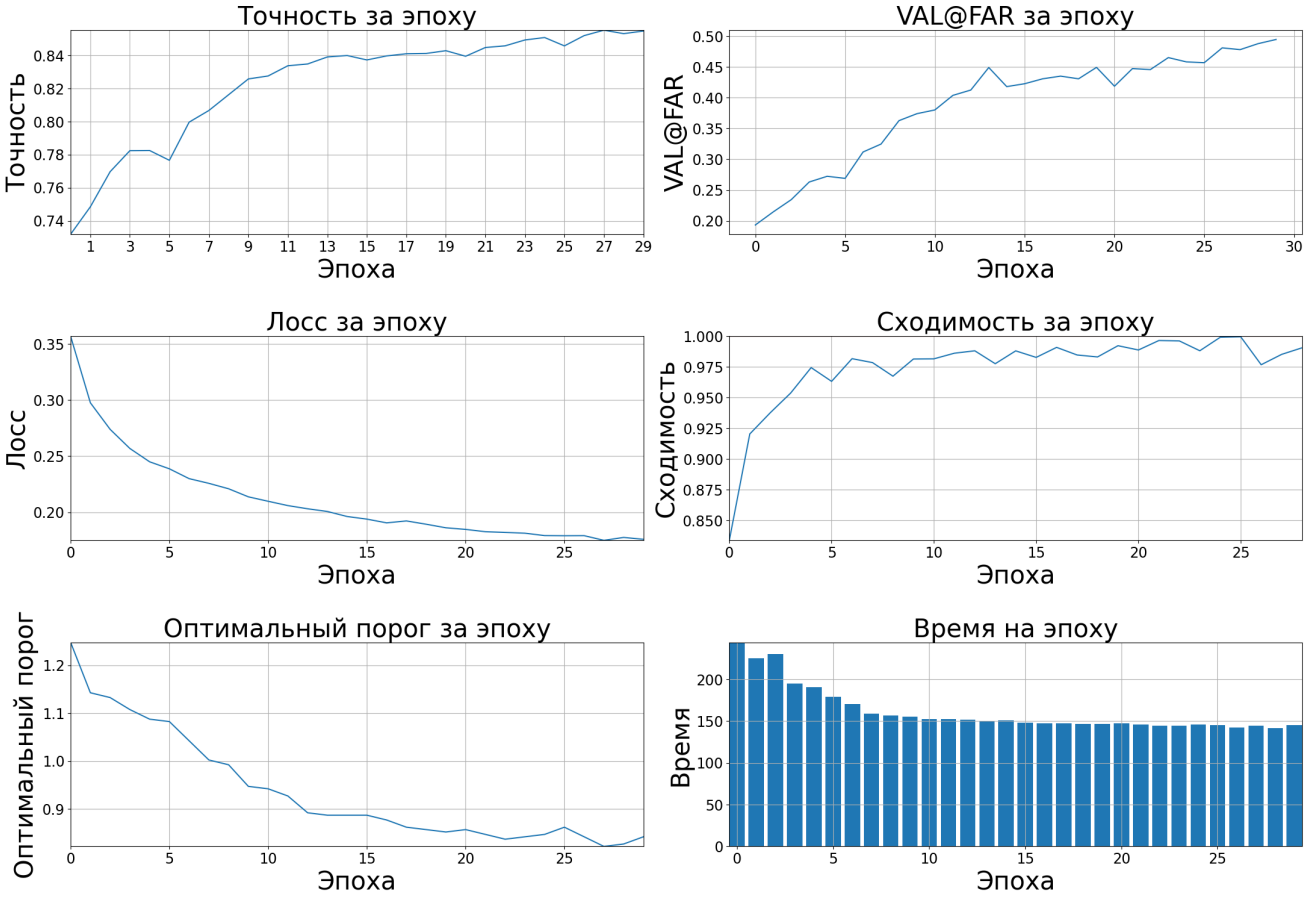

Рис. 3: CS-Loss

Точность	VAL@FAR(10^{-2})	Порог (threshold)	Среднее время на эпоху
0.86	0.48	0.82	162

Из графиков можно сделать вывод о том, что обучение вначале было медленнее, чем в конце, что можно связать с тем, что приходится считать больше градиентов так как многие вектора в кластере находятся дальше, чем пороговое значение. Обучение стабильно, видно, что метрики сильно коррелируют с лоссом, что еще раз подтверждает эффективность функции потерь.

Обучение с Triplet loss random mining

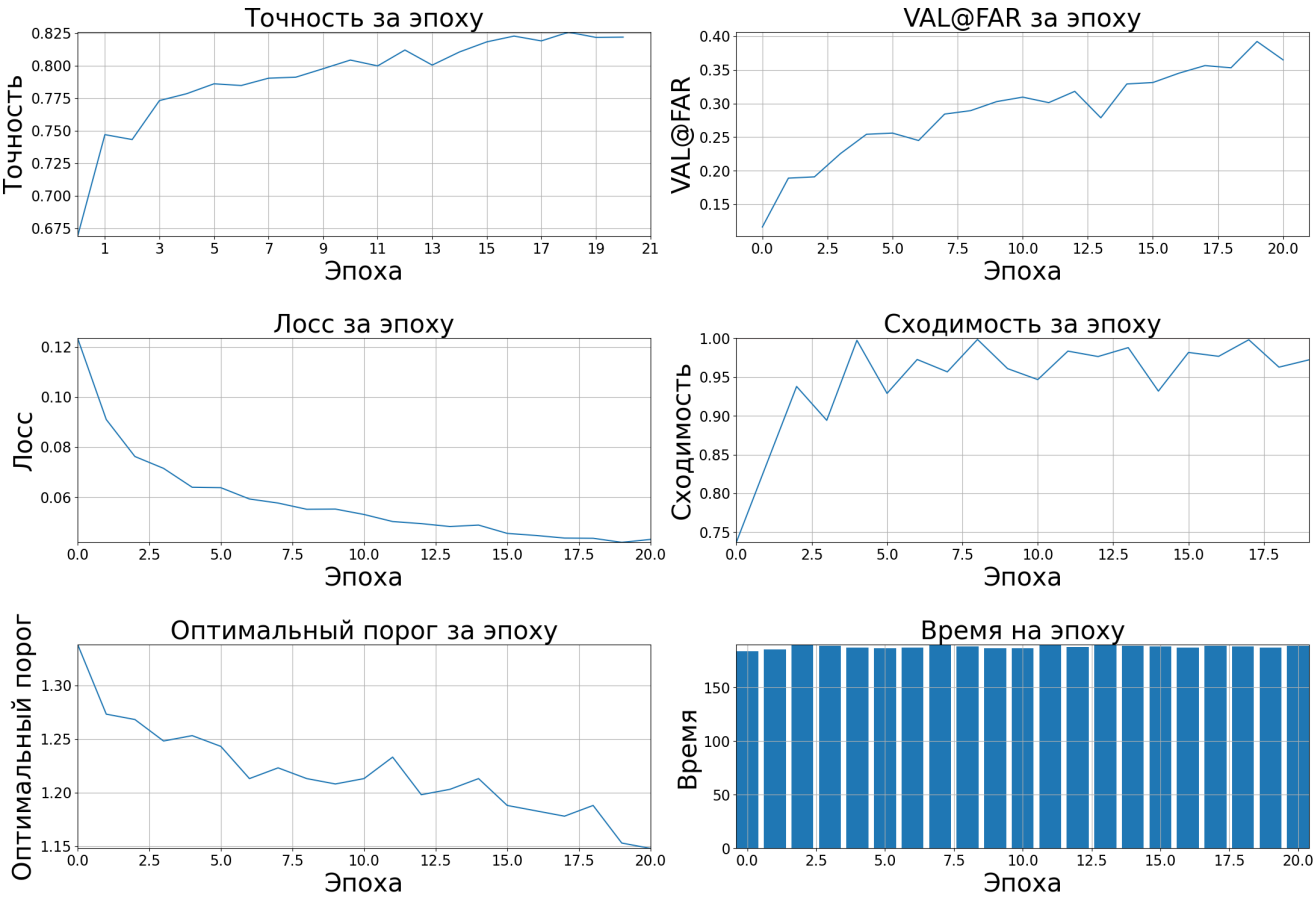

Рис. 4: Triplet loss со случайными триплетами

Точность	VAL@FAR(10^{-2})	Порог (threshold)	Среднее время на эпоху
0.83	0.35	1.19	188

Рандомный майнинг показал себя лучше всех по сравнению с другими методами майнинга с точки зрения максимальных значений метрик. Связать это можно с тем, что на заключительных этапах рандомный майнинг, в отличие от полусложного и сложного майнинга не берет некачественные данные в приоритете (как раз два других метода майнинга из-за принципа своей работы будут искать некачественные картинки, ибо именно среди них будут далеки позитивные примеры, например при плохом качестве фотографии или ракурсе съемки).

Обучение с Triplet loss semi-hard mining

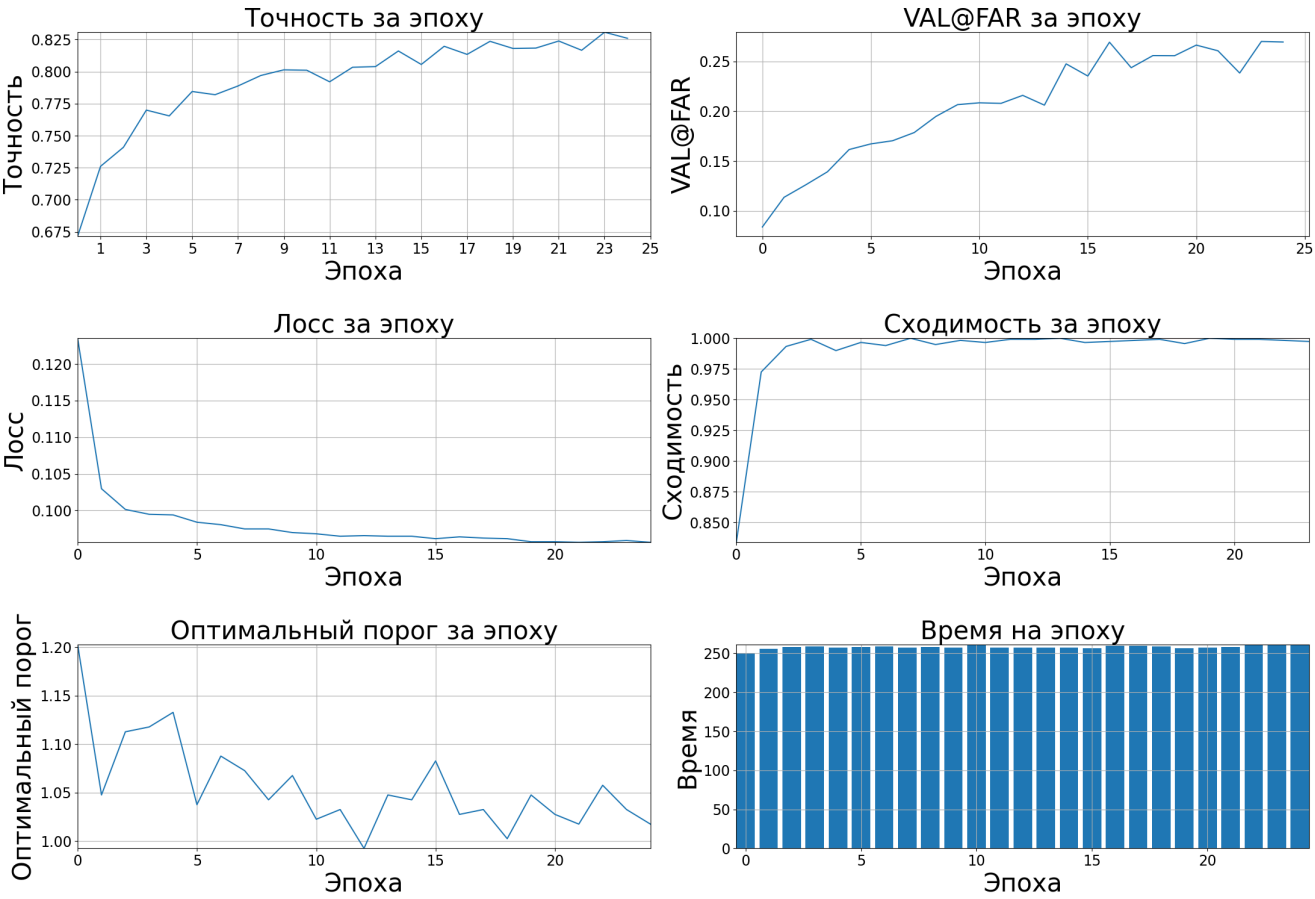

Рис. 5: Triplet loss с полусложными триплетами

Точность	VAL@FAR(10^{-2})	Порог (threshold)	Среднее время на эпоху
0.83	0.27	1.03	258

Полусложный майнинг показал себя чуть хуже, чем рандомный по тем причинам, что я уже указывал, но при этом занимает намного более значительное время для обучения в связи с необходимостью искать позитивный и негативный пример, подходящий под условия.

Обучение с Triplet loss hard mining

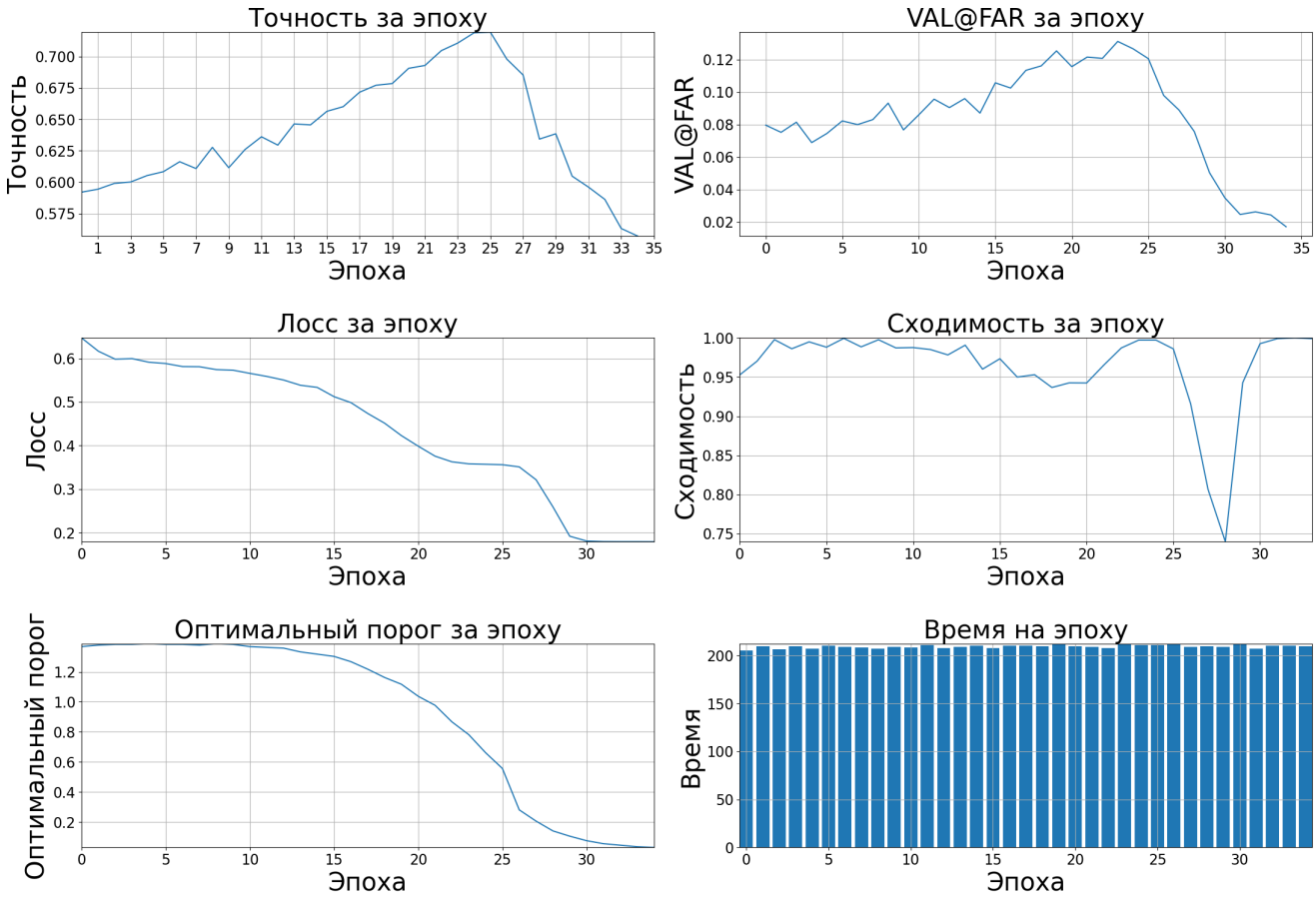

Рис. 6: Triplet loss со сложными триплетами

Точность	VAL@FAR(10^{-2})	Порог (threshold)	Среднее время на эпоху
0.72	0.12	0.56	209

Сложный майнинг оказался хуже всех остальных и вовсе не сошелся в локальный минимум (в константу), так как в какой-то момент, среди самых дальних позитивных примеров кол-во некачественных фотографий стало критическим и модели оказалось проще переводить все изображения в константу.

Выводы

1. Введённый в данной работе `CS-Loss` показал лучшие результаты из представленных, как по значениям метрик, так и по скорости работы. Помимо этого, он продемонстрировал стабильность и простоту применения.

2. `CS-Loss` прекрасно работает "из коробки", не требуя дополнительной настройки стратегий майнинга, в отличие от `triplet loss`.

3. `Random mining` на удивление показал лучший результат среди способов майнинга триплетов, что идёт вразрез с результатами, представленными в работе [1].

4. Модель, обученная на hard триплетах, не сошлась вопреки теоретическим ожиданиям. Возможными причинами могли быть использование слишком маленького размера батча (увеличение размера батча могло бы привести к улучшению результатов), а также низкое качество датасета (проблема могла бы быть исправлена ручной проверкой и удалением "плохих" изображений, например, низкого качества).

Возможные улучшения CS-Loss

Для повышения эффективности новой функции потерь (*CS-Loss*) можно рассмотреть следующие направления:

Пересмотр понятия центра временного кластера

- Использование адаптивных методов вычисления центра кластера, например, экспоненциального скользящего среднего (EMA).
- Учет не только пространственной близости, но и временной динамики изменения центров.
- Введение весов для объектов кластера в зависимости от их "возраста"или степени уверенности.

Автоматизация подбора гиперпараметров

- Применение методов байесовской оптимизации для автоматического выбора оптимальных параметров в процессе обучения.
- Использование адаптивных стратегий, подобных *learning rate scheduling*, но для параметров кластеризации.
- Внедрение механизмов мета-обучения (meta-learning) для настройки гиперпараметров на лету.

Детектирование вбросов

- Интеграция статистических методов (например, анализ межквартильных размахов) для выбросов.
- Использование методов, основанных на плотности (DBSCAN-like подходы) для игнорирования шумовых точек.

Список литературы

[1] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[2] Marcelo Cicconet, H Elliott, David L Richmond, D Wainstock, and M Walsh. Image forensics: detecting duplication of scientific images with manipulation-invariant image similarity. *arXiv preprint arXiv:1802.06515*, 2018.

[3] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, pages 1–30. Lille, 2015.

[4] Baoyang Song. Deep neural network for learning to rank query-text pairs. *arXiv preprint arXiv:1802.08988*, 2018.

[5] Tharindu Ranasinghe, Constantin Orăsan, and Ruslan Mitkov. Semantic textual similarity with siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1004–1011, 2019.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[7] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019.

[8] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[9] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[10] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017.

[11] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018.