МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теории функций и стохастического анализа

# РЕШЕНИЕ ЗАДАЧ РАСПОЗНАВАНИЯ ОБРАЗОВ

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

студента 2 курса 248 группы
направления 09.04.03 — Прикладная информатика

механико-математического факультета
Емельянова Тимофея Дмитриевича

Научный руководитель
доцент, к. ф.-м. н., доцент _____ М. Г. Плешаков

Заведующий кафедрой
д. ф.-м. н., доцент _____ С. П. Сидоров

Саратов 2022

# ВВЕДЕНИЕ

Данная работа посвящена решению задач распознавания образов. Методы распознавания образов являются разделом теории искусственного интеллекта, в котором решают задачи, связанные с классификацией и идентификацией объектов и сигналов различного происхождения.

На сегодняшний день можно выделить два основных направления в области распознавания образов:

— изучение способностей к распознаванию, которыми обладают живые существа, моделирование и объяснение их;

— развитие теории и методов построения устройств, предназначенных для решения отдельных задач в прикладных целях.

Для решения задач распознавания образов в работе используются искусственные нейронные сети, которые часто применяют для решения задач не только компьютерного зрения, но и других задач, связанных с оптимизацией, прогнозированием, обработкой естественного языка и др. В практической части будет показано, как решить задачу распознавания образов в задачах компьютерного зрения на примере поиска похожих картин с помощью нейронных сетей.

**Работа актуальна**, поскольку распознавание образов — относительно новая область искусственного интеллекта, но несмотря на это, уже сейчас имеются многочисленные важные приложения методов распознавания во многих сферах науки, техники, медицине, биологии и т.д.

**Целью работы** является решение различных задач компьютерного зрения. А в качестве **средства** будут использоваться искусственные нейронные сети как эффективный метод.

Для достижения цели были поставлены следующие **задачи**:

— изучить основные определения, применяемые в искусственных нейронных сетях;

— исследовать современные подходы к решению задач компьютерного зрения, а также недостатки этих подходов;

— рассмотреть архитектуры искусственных нейронных сетей;

— познакомиться с библиотекой машинного обучения pytorch;

— научиться применять сверточные нейронные сети для решения задач компьютерного зрения.

В работе большое внимание уделено практической части, поскольку решается сразу несколько задач компьютерного зрения: классификация, регрессия и поиск похожих изображений. **Практическая значимость** работы состоит в том, что в дальнейшем полученные результаты и модели можно использовать для решения других задач распознавания образов.

Работа прошла апробацию на различных конференциях, в частности, на ежегодной студенческой конференции "Актуальные проблемы математики и механики", которую проводил механико-математический факультет СГУ в апреле 2022 года, в секции "Анализ данных", в X Международной научно-практической конференции «Математическое и компьютерное моделирование в экономике, страховании и управлении рисками», ноябрь 2021 года.

**Структура и содержание магистерской работы.** Работа состоит из введения, трех разделов, заключения, списка использованных источников, содержащего 24 наименования. Общий объем работы составляет 51 страницу.

## ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во **введении** обосновывается актуальность работы, формулируются цель и решаемые задачи.

В **первом** разделе даются основные понятия из теории искусственных нейронный сетей: искусственный нейрон, полносвязная сеть, приводятся примеры различных функций активации и функций потерь для задач регрессии и классификации. Также в разделе описывается процесс обучения искусственной нейронной сети.

*Искусственный нейрон* — это базовый элемент искусственной нейронной сети. Он является упрощенной моделью естественного нейрона в нервной системе человека.

Искусственный нейрон состоит из следующих элементов:

— вектор входных значений $\vec{X} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$,

— веса $\vec{W} = (w_1, w_2, \ldots, w_n) \in \mathbb{R}^n$,

— смещение $b \in \mathbb{R}$,

— нелинейная функция активации $f$, зависящая от одного аргумента,

— выход нейрона $y \in \mathbb{R}$.

На рисунке 1 приведено строение одного нейрона.

Учитывая обозначения выше, математически нейрон можно описать следующей формулой:

$$y = f\left(\sum_{i=1}^{n} x_i w_i + b\right). \tag{1}$$

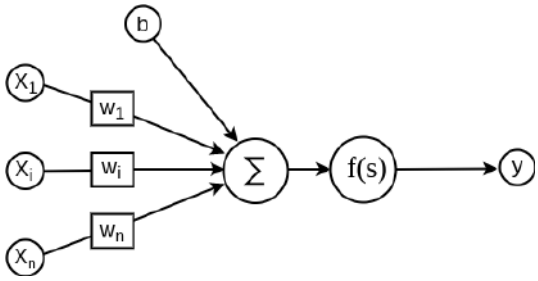


Рисунок 1 – Нейрон

*Многослойные искусственные нейронные сети* состоят из нейронов, которые делятся на группы с общим входным сигналом. Каждую такую группу называют *слоем* нейронной сети. Внешняя информация поступает на входной слой нейронной сети. *Входной слой* — это первый слой в нейронной сети, который принимает входящие сигналы и передает их на последующие уровни. Входной слой часто считают нулевым. После этого выходы входного слоя поступают на скрытые слои. Выходами нейронной сети являются выходные сигналы последнего, т.е. *выходного* слоя.

Искусственная нейронная сеть называется *полносвязной*, если все выходы нейронов $i$-го слоя передаются на вход каждого нейрона $(i+1)$-го слоя.

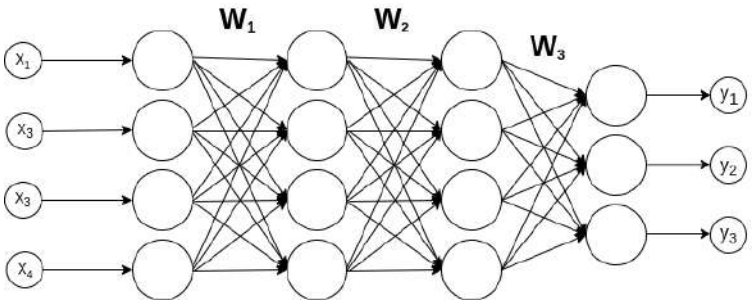


Рисунок 2 – Пример нейронной сети

На рисунке 2 приведено построение многослойной нейронной сети, ко-

торая принимает на вход вектор $\vec{X}$ размерности 4, имеет два скрытых слоя и выходом которой является вектор $\vec{Y}$ размерности 3.

Во **втором** разделе разобраны недостатки применения полносвязных нейронных сетей к задачам, связанным с изображениями, а также показаны другие архитектуры сетей, которые могут эффективно решать задачи распознавания образов.

Черно-белое изображение в компьютере может быть представлено в виде двумерной матрицы чисел, где каждое число принадлежит отрезку $[0, 255]$. 0 соответствует черному пикселю, а 255 — белому. Пример представления изображения в компьютере можно увидеть на рисунке 3.

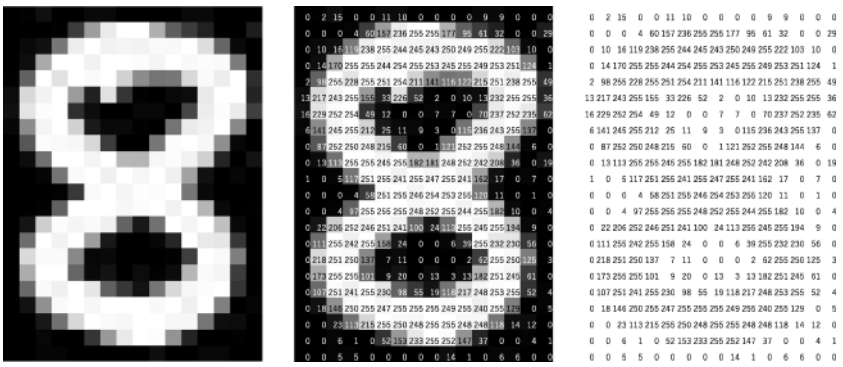

Рисунок 3 – Представление изображения в компьютере

Допустим, что необходимо решить задачу классификации цифр на изображении. Для такой задачи может быть использована полносвязная нейронная сеть, однако для этого необходимо преобразовать входные данные в вектор. Если изначальное изображение представлено матрицей $\mathbf{M}$ размерности $w \times h$, то в качестве входного вектора может быть использован вектор $\vec{V}$ размерности $1 \times w \cdot h$, который состоит из последовательно объединенных строк матрицы $\mathbf{M}$. Таким образом, получается $w \cdot h$ входных значений.

Недостатки такого подхода:

— очень много нейронов во входном слое сети. Например, если изображение имеет разрешение $256 \times 256$ пикселей, то входной слой сети будет состоять из $256 \cdot 256 = 65536$ нейронов. Это приводит к тому, что такую сеть тяжело обучать;

— не учитываются пространственные отношения на изображении, которые могли бы помочь сети эффективнее решать задачу.

*Фильтр (ядро)* — квадратная матрица размерности $k \times k$. Фильтры используются для операции свертки. Если проводить аналогию с полносвязными сетями, то фильтры — это веса, которые необходимо настраивать в процессе обучения нейронной сети.

*Операция свертки* — это некоторое линейное преобразование входных данных, которое выполняется с помощью фильтра.

Операция свертки производится над двумя матрицами $\mathbf{M}$ (исходная матрица) и $\mathbf{K}$ (фильтр) размера $w \times h$ и $k \times k$ соответственно.

Чтобы применить операцию свертки к изображению, которое может быть представлено в виде матрицы $\mathbf{M}$, необходимо поэтапно скалярно умножать значения матрицы на фильтр. Результаты будут записываться в отдельную матрицу, которая носит название *матрица признаков* (или карта признаков), будем обозначать ее $\mathbf{C}$.

В виде формулы операция свертки может быть записана так:

$$C_{i,j} = \sum_{u=0}^{k-1} \sum_{v=0}^{k-1} M_{i+u,j+v} K_{u,v}. \tag{2}$$

На рисунке 4 рассмотрим применение операции свертки с заданным фильтром размерности $3 \times 3$ к исходной матрице размерности $5 \times 5$, в результате чего получается матрица признаков $3 \times 3$.

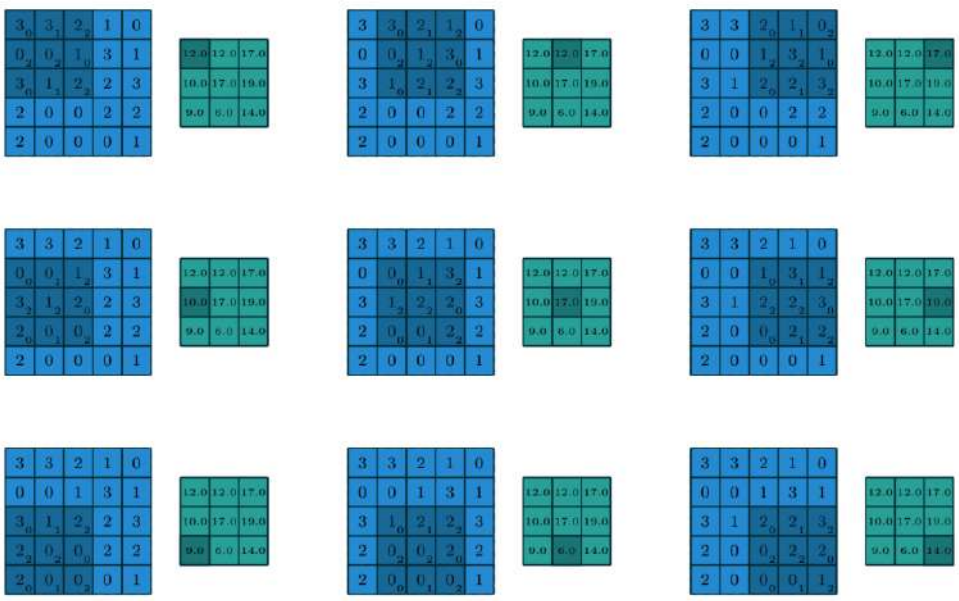

Рисунок 4 – Пример выполнения операции свертки

В **третьей** части работы описано применение искусственных нейронных сетей для решения различных задач компьютерного зрения, а именно: задачи классификации, задачи регрессии и задачи поиска похожих изображений.

В качестве **первой** задачи рассмотрим следующую проблему: представьте, что вы хотите понять, была ли написана выбранная вами гравюра определенным гравером. В качестве автора был выбран один из самых популярных граверов — Альбрехт Дюрер, который за свою жизнь написал более 200 различных гравюр.

Для формирования набора данных для обучения сети использовались изображения из открытого источника — электронного каталога немецкой гравюры XV–XX веков ГМИИ им. А.С. Пушкина ( `http://germanprints.ru/`).

Данная задача является задачей бинарной классификации. В работе показано, как можно эффективно решить ее даже с небольшим количеством данных. На рисунке 5 можно увидеть архитектуру сети.

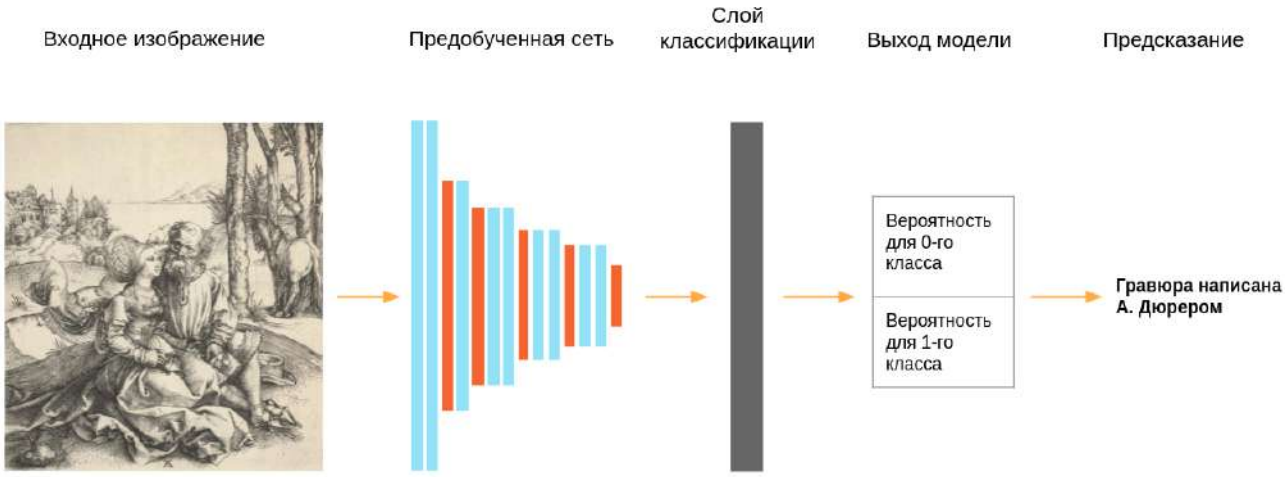


Рисунок 5 – Архитектура сети

Оценить качество работы полученной модели можно с помощью матрицы ошибок для тестового набора данных и предсказания модели 6.

Для демонстрации работы модели было написано приложение на языке Python с использованием библиотеки Flask. На рисунке 7 можно увидеть результат работы приложения для оригинальной гравюры Дюрера и копии.

Заметим, что в случае классического подхода к классификации изображений имеются некоторые ограничения и условия:

| | | Предсказание | |
|---|---|---|---|
| | класс | 0 | 1 |
| Реальность | 0 | **56** | **3** |
| | 1 | **2** | **27** |

Рисунок 6 – Матрица ошибок для тестового набора данных

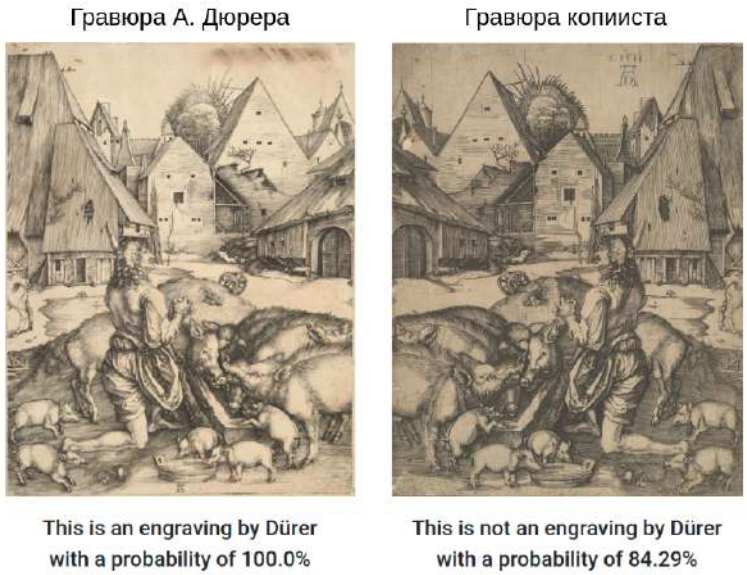


Рисунок 7 – Пример работы модели классификации гравюр А. Дюрера

— набор данных должен содержать большое количество примеров для каждого класса;
— с увеличением количества классов быстро будет возрастать ошибка;
— чтобы добавить в модель новый класс, необходимо переобучать всю модель.

**Вторая** задача в работе относится к классу задач, которая не может быть эффективно решена с помощью классического подхода. Этой задачей является поиск похожих картин.

*Embedding-вектор* — это вектор, который будет описывать исходное изображение в многомерном пространстве заданной размерности. В задаче будет использован вектор размерности 128. Задача поиска похожих объектов состоит в том, чтобы расстояние между векторами объектов из одного класса было маленьким, а между векторами объектов из разных классов — большим.

*Embedding-слой* — это полносвязный слой, который следует сразу после выхода предобученной нейронной сети. Embedding-слой является тем, что отличает данную архитектуру от классической классификации. Размерность этого слоя совпадает с размерностью Embedding-вектора.

*Слой классификации* — это полносвязный слой, который следует за Embedding-слоем. Этот слой используется, чтобы из Embedding-вектора получить вектор вероятностей для классов.

Архитектура сети для решения задачи поиска похожих картин представлена на рисунке 8.

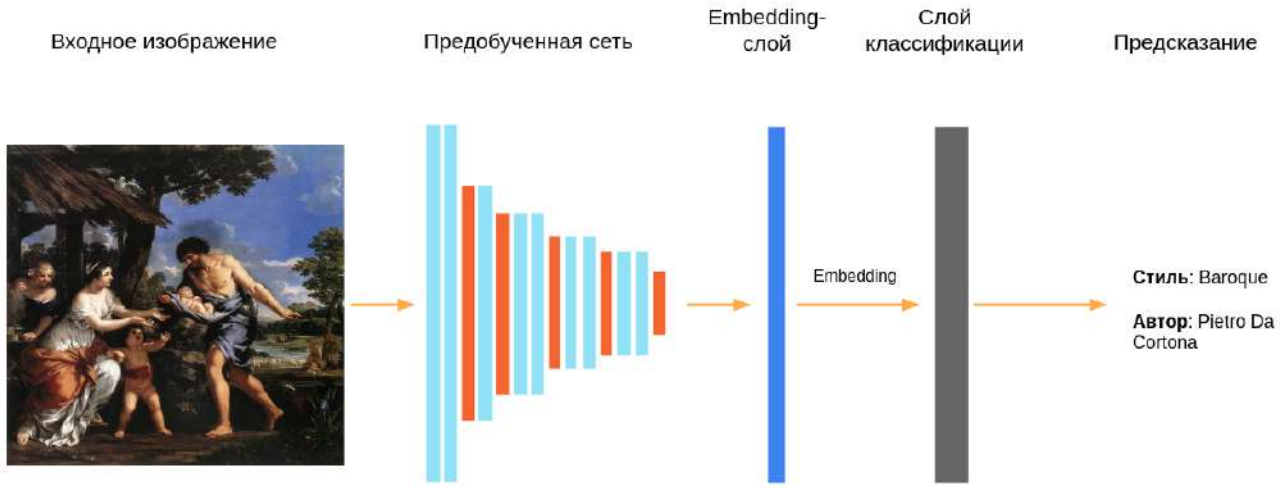


Рисунок 8 – Архитектура сети

Веса в слое классификации, очевидно, являются двумерной матрицей, где первая размерность соответствует размерности Embedding-вектора (будем использовать вектор размерности 128), а вторая — количеству классов. Это значит, что для каждого класса имеется некий вектор (строка в матрице весов для слоя классификации), размерность которого равна размерности Embedding-вектора. Такой вектор принято называть *центроидом* класса.

Тогда обучение сети сводится к получению таких весов в Embedding-слое, чтобы вектор, который получается после выхода слоя, был ближе всего к центроиду класса, которому соответствует изначальная картина. Важно учитывать, что в процесса обучения изменяются веса и Embedding-слоя и слоя классификации (это значит, что центроиды не фиксированы).

В качестве источника данных был использован сайт `https://www.wikiart.org/`, который помимо самих картин, также содержит

различную информацию о картинах, например, автора, стиль, год написания и др.

Для обучения модели используются работы 1119 авторов в 27 стилях — это около 80000 изображений.

В качестве класса будем использовать пересечение автора картины и стиля, в котором данная картина написана, поскольку именно такие картины являются наиболее похожими между собой. Набор данных содержит 2108 таких классов.

Когда модель обучена, можно убрать последний слой классификации и использовать выход после Embedding-слоя. Сначала необходимо получить Embedding-векторы для всех изображений, которые имеются в нашем наборе данных, а затем сохранить их. Когда пользователь захочет для своего изображения найти похожие изображения из имеющегося у нас набора данных, необходимо просто получить векторное представление этого изображения с помощью модели, а затем найти ближайшие вектора в имеющемся наборе данных и вернуть пользователю картины, которые соответствуют этим векторам.

Описание работы сервиса можно увидеть на рисунке 9.

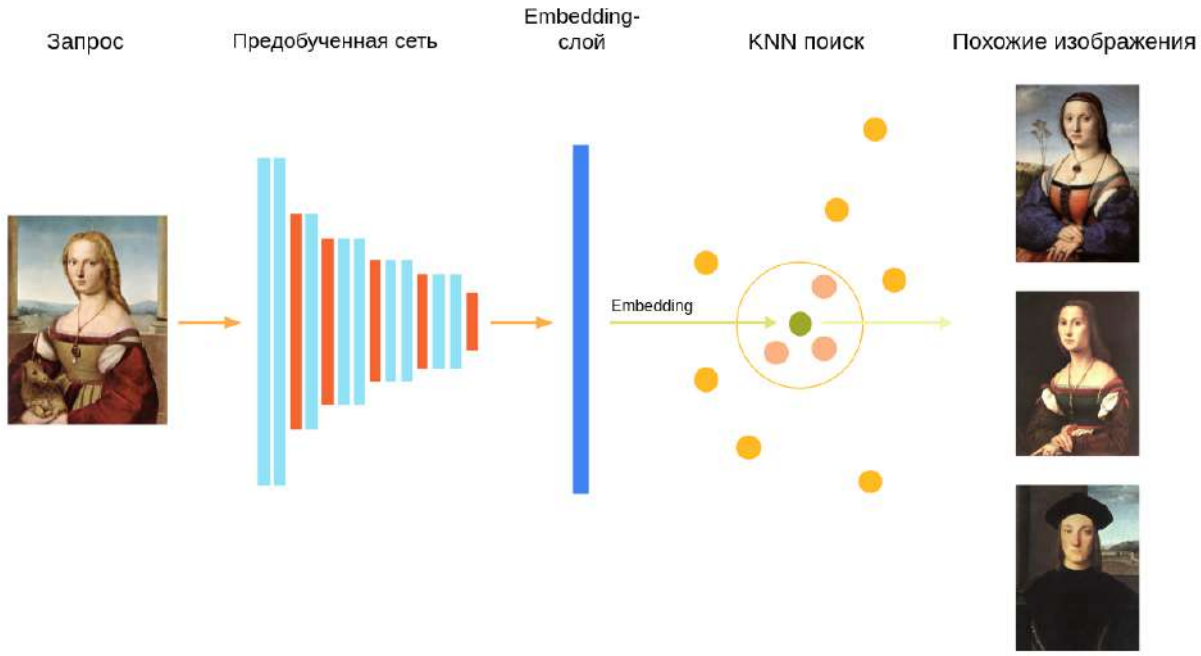


Рисунок 9 – Процесс поиска похожих картин

В качестве основной метрики используется $1-Recall@k$, где $k = 1, 2, 5, 10, 20$.

Эта метрика подсчитывает долю предсказаний, где хотя бы один верный класс присутствует среди $k$ первых результатов поиска. Например $1-Recall@5 = 0.6$ значит, что в 60% случаев хотя бы одна картина верного класса была найдена среди 5 первых «наиболее похожих» картин, которые вернула модель.

Результаты можно увидеть на рисунке 10.

| | k=1 | k=2 | k=5 | k=10 | k=20 |
|---|---|---|---|---|---|
| Softmax | 0.438 | 0.477 | 0.492 | 0.534 | 0.596 |
| ArcFace (s=25, m=0.3) | **0.441** | **0.503** | **0.582** | **0.636** | **0.687** |

Рисунок 10 – значения 1-Recall@k метрики

Помимо стиля и автора для многих картин также известен год написания. Таким образом, в **третьей** задаче показано, как построить алгоритм для предсказаний этого значения.

Осуществлять поиск похожих картин и определять год написания будем в одном приложении, поэтому обучать отдельную нейронную сеть для этой задачи не является рациональным решением, поскольку в таком случае практически вдвое увеличивается время работы сервиса.

Первое, что можно попробовать — это использовать среднее или медианное значение для годов написания похожих картин, которые получаются при решении задачи поиска похожих картин. В работе описывается, почему такое решение хорошо себя показывает.

Другой подход к решению этой задачи без тренировки новой полноценной модели — это добавление новых полносвязных слоев после *Embedding*-слоя к модели, которая уже обучена находить похожие картины. На рисунке 11 можно увидеть архитектуру такой модели. В качестве новых слоев были добавлены 3 полносвязных слоя с количеством выходов 256, 64 и 1.

Во время тренировки модели изменялись веса только новых добавленных слоев.

Для оценки работы моделей будем использовать две метрики: **RMSE** (root mean squared error) и **MAE** (mean absolute error):

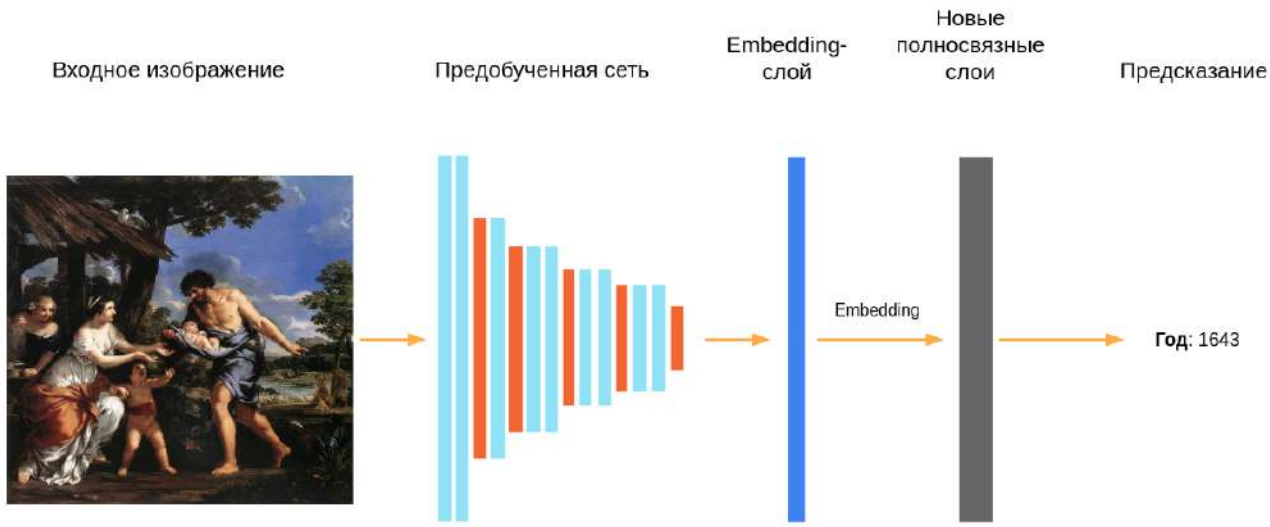


Рисунок 11 – Архитектура модели для определения года написания картины

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}, \tag{3}$$

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|, \tag{4}$$

где $N$ — размер батча, $y_i$ — предсказание модели для $i$-го элемента в батче, $\hat{y}_i$ — истинное значение для $i$-го элемента в батче.

Результаты можно увидеть на рисунке 12

| Метрика / Модель | **RMSE** | **MAE** |
|---|---|---|
| Среднее для 20 похожих картин | 29.6 | 13.39 |
| Обучение новых слоев | 27.35 | 13.32 |

Рисунок 12 – Метрики работы моделей для определения года написания картин

Для демонстрации работы моделей было написано приложение на Python с использованием библиотеки Flask. Пример работы можно увидеть на рисунке 13 (загруженное изображение находится сверху, снизу — похожие изображения, которые возвращает модель поиска похожих, а также в самом верху можно увидеть предсказанный год написания картины).

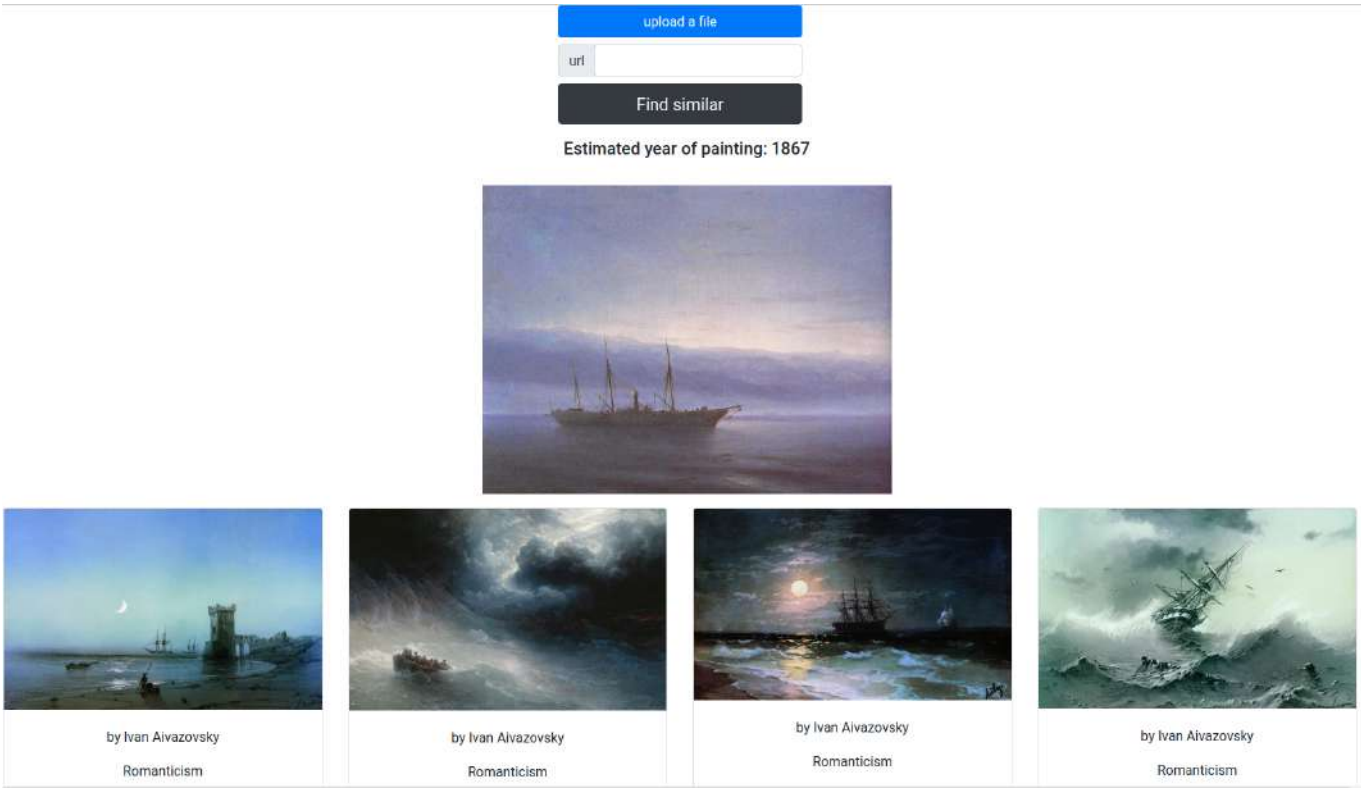


Рисунок 13 – Пример работы модели (Иван Константинович Айвазовский, 1872)

## ОСНОВНЫЕ РЕЗУЛЬТАТЫ

В ходе работы были определены основные понятия и обозначения, применяемые в теории искусственных нейронных сетей, а также изучены современные подходы к решению задач компьютерного зрения.

В рамках работы были изучены архитектуры полносвязных и сверточных нейронных сетей, а также рассмотрены недостатки и преимущества этих архитектур.

В ходе выполнения практической части работы были написаны все необходимые программы для решения задачи классификации, регрессии и поиска похожих картин. Также были написаны приложения для отображения работы полученных моделей. Ввиду большого количества кода, был создан открытый репозиторий на платформе Github. Репозиторий располагается по ссылке `https://github.com/Juravlik/ssu_artworks`

Полученные нейронные сети в дальнейшем могут быть использованы и для решения других задач компьютерного зрения.