Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования
«Севастопольский государственный университет»

Методические указания
к выполнению итогового проекта
по дополнительной профессиональной программе
профессиональной переподготовки
“Глубокие нейросети в компьютерном зрении”

Севастополь

2024

УДК 004.8 (075.8)

Методические указания к выполнению итогового проекта по дополнительной профессиональной программе профессиональной переподготовки “Глубокие нейросети в компьютерном зрении”/ СевГУ; сост. **В. Н. Бондарев**. **—** Севастополь: Изд-во СевГУ, 2024. **—**26 с.

Методические указания предназначены для использования при выполнении итогового проекта по дополнительной профессиональной программе профессиональной переподготовки “Глубокие нейросети в компьютерном зрении”. Излагаются цели, задачи, этапы проектирования глубоких нейросетей, рассматриваются типовые задачи компьютерного зрения, приводятся ссылки на доступные датасеты и литературные источники с архитектурами нейросетей, рассматривается пример итогового проекта.

Методические указания рассмотрены и утверждены на заседании кафедры «Информационные системы» (протокол № от февраль 2024 г.)

Рецензент: Брюховецкий А.А. , к.т.н., доцент кафедры информационных технологий и компьютерных систем

СОДЕРЖАНИЕ

Введение

Обучение по дополнительной профессиональной программе профессиональной переподготовки «Глубокие нейросети в компьютерном зрении» завершается подготовкой командного/индивидуального проекта в среде Python/TensorFlow/Keras/Pytorch».

Цель проекта состоит в том, чтобы слушатели самостоятельно реализовали весь конвейер глубинного обучения: загрузку данных, проектирование модели, обучение, оценку и анализ.

В ходе итоговой аттестации предусматривается защита проекта с представлением презентации проекта.

1. Выбор темы итогового проекта

Итоговый проект — это возможность применить те знания и навыки, которые вы получили в ходе обучения по Программе «Глубокие нейросети в компьютерном зрении», к интересующей вас предметной области. Потенциальные проекты делятся на два возможных направления:

1. *Проекты, ориентированные на разработку приложения*. Если у вас имеется конкретная область интересов, то выберите реальную задачу из этой области и примените нейросетевые модели компьютерного зрения, изученные на Программе, к решению иыбранной задачи;
2. *Проекты, ориентированные на разработку модели нейросети*. Вы можете построить новую нейросетевую модель или новый вариант существующей модели и применить ее для решения задач компьютерного зрения. Этот вариант проекта может быть более сложным и может быть сопряжен с подготовкой дальнейшей научной работы.

Следует отметить одно ограничение при выборе темы проекта. Программа «Глубокие нейросети в компьютерном зрении» посвящена компьютерному зрению, поэтому ваш проект должен включать обработку пикселей визуальных данных в той или иной форме. Например, чистый проект по обработке естественного языка — не лучший выбор, даже если ваш подход предполагает использование сверточных сетей.

Чтобы вдохновиться некоторыми идеями для выбора темы проекта, можно просмотреть тематику и публикации трудов конференций по компьютерному зрению и глубокому обучению, а также другие ресурсы и книги указанные ниже [1-7]:

- CVPR: тьюториалы IEEE конференции по компьютерному зрению и распознаванию образов, а также некоторые труды конференции;
- WACV: труды IEEE/CVF Зимней конференции по приложениям компьютерного зрения (Winter Conference on Applications of Computer Vision - WACV);

- ICCV: тьюториалы Международной конференции по компьютерному зрению
- ECCV: тьюториалы Европейской конференции по компьютерному зрению;
- NeurIPS: ресурс, включающий журнал по нейронным системам обработки информации и описание датасетов и бенчмарков.

Вы также можете получить представления относительно возможной темы проекта, взглянув на некоторые популярные наборы данных компьютерного зрения по следующим ссылкам:

- мета указатель на базы данных для задач компьютерного зрения CV Datasets on the web;

- мета указатель на базы данных для задач компьютерного зрения YACVID – Yet Another Computer Vision Index to Datasets;

- мета указатель на задачи и базы данных для обучения нейросетей, включая задачи и базы данных компьютерного зрения Browse State-of-the-Art;

- ImageNet: датасет изображений большого масштаба для визуального распознавания в соответствии с WordNet иерархией;

- Places datasets: набор данных для высоуровневого понимания изображений: распознавание сцен, обнаружение объектов, предсказание действий и событий;

- Places Database: база данных для анализа сцен, содержащая 205 категорий сцен и 2.5 миллиона размеченных изображений;

- NYU Depth Dataset v2: RGB-D датасет сегментированных внутренних помещений;

- Microsoft COCO: датасет для распознавания изображений, сегментации и формирования подписей к изображениям;

- Flickr100M: 100 миллионов изображений с Flickr;

- Labeled Faces in the Wild: датасет, содержащий 13000 фотографий лиц с метками;

- CelebFaces Attributes: коллекция содержит более 200000 цветных изображений лиц знаменитостей, снабженных категориями принадлежности (например, *в шляпе*, *с улыбкой* и т. д.);

- Fashion-MNIST: набор данных, состоящий из 70 000 изображений 28×28 в оттенках серого модных товаров из 10 категорий, по 7000 изображений в каждой категории;

- LSUN: коллекция данных для классификации содержит 10 категорий сцен, таких как спальни, кухни, столовые, конференц-залы, церкви и др. Каждая категория содержит огромное количество изображений — от 120000 до 3000000;

- Human Pose Dataset: набор данных для оценки шарнирно представленной позы человека;

- YouTube Faces DB: видео датасет для распознавания лиц на видео;

- UCF101: датасет для распознавания действий на реалистичных видео, содержащий 101 категорию;

- HMDB-51: большой датасет с движениями человека, содержащий 51 класс;
- ActivityNet: большой видео датасет для распознавания действий человека;
- Moments in Time: большой датасет для распознавания и понимания действий и событий на видео, содержащий 1 миллион 3-х секундных видео.

Проекты, ориентированные на решение задач компьютерного зрения, часто используют сверточные сети. Проекты такого типа предполагают понимание современных нейронных моделей компьютерного зрения, а также создание новых моделей или улучшение существующих моделей. В ходе реализации проектов осуществляется тщательная подготовка данных, определяются соответствующие функции потерь, детали обучения и кросс-валидации, а также выполняется оценка разработанных моделей на тестовых наборах и осуществляется сопоставление моделей.

Сверточные сети могут также работать в режиме реального времени на мобильных телефонах и Raspberry Pi . Создание интересного мобильного приложения могло бы стать хорошим проектом. Если вы хотите пойти по этому пути, возможно, вы захотите воспользоваться версиями фреймворков для мобильных приложений: TensorFlow Mobile/Lite или Caffe2 /Android [7].

В списке ниже представлены возможные тематические направления проектов компьютерного зрения для первоначального выбора:

1. Классификация изображений;
2. Сегментация изображений;
3. Обнаружение объектов на изображении;
4. Классификация сцен;
5. Обнаружение лиц;
6. Распознавание эмоций по изображениям лиц;
7. Формирование заголовков к изображениям;
8. Аугментация (расширение) изображений;
9. Анализ аудио или речи с помощью CNN;
10. Визуализация состояний нейросетей;
11. Стилизация изображений;
12. Генерация изображений.

Ниже приведено описание некоторых из указанных задач компьютерного зрения.

Классификация изображений — это фундаментальная задача распознавания изображений, цель которой приписывание изображения одному из заданных классов с использованием меток изображений. В отличие от обнаружения объектов, которое включает в себя классификацию и определение местоположения нескольких объектов на изображении, классификация изображений обычно связана с изображением одного объекта.

Семантическая сегментация — это задача компьютерного зрения, цель которой — отнести каждый пиксель изображения к классу или объекту. Цель состоит в том, чтобы создать карту сегментации изображения по пикселям, где

каждый пиксель приписывается определенному классу или объекту. Примерами датасетов для этой задачи являются Cityscapes, PASCAL VOC и ADE20K. Модели обычно оцениваются с помощью показателей Mean Intersection-Over-Union (Mean IoU) и Pixel Accuracy. Кроме семантической сегментации, выделяют задачи сегментации экземпляров, паноптической сегментации.

Обнаружение объектов на изображении — это задача компьютерного зрения, цель которой обнаружить и найти интересующие объекты на изображении. Задача включает в себя определение положения и границ объектов на изображении, а также классификацию объектов по различным категориям. Она является важной частью распознавания изображений наряду с классификацией и поиском изображений. Современные методы можно разделить на два основных типа: одноэтапные методы и двухэтапные методы. В одноэтапных методах приоритет отдается скорости обработки и принятия решения. Примеры моделей включают YOLO, SSD и RetinaNet. В двухэтапных методах приоритет отдается точности обнаружения. Примеры моделей включают Faster R-CNN, Mask R-CNN и Cascade R-CNN. Самым популярным датасетом является набор данных MS COCO. Модели обычно оцениваются по показателю средней точности.

Аугментация (расширение) изображений — это метод увеличения данных, который генерирует больше обучающих данных из существующих обучающих выборок. Аугментация изображений особенно полезна в областях, где обучающие данные ограничены или их получение дорого, например, в биомедицинских приложениях.

Обнаружение краев — это фундаментальный метод обработки изображений, который включает в себя вычисление градиента изображения для количественной оценки величины и направления краев и границ на изображениях. Градиенты изображения используются в различных последующих задачах компьютерного зрения, таких как обнаружение линий, обнаружение признаков и классификации изображений. Границы и края на изображениях можно выделять сверточными слоями.

Обнаружение лиц — это задача компьютерного зрения, которая включает в себя автоматическую идентификацию и обнаружение человеческих лиц на цифровых изображениях или видео. Это фундаментальная технология, лежащая в основе многих приложений, таких как распознавание лиц, отслеживание лиц и анализ лиц.

Преобразование (перенос) стилей — это метод компьютерного зрения, который предполагает создание нового изображения путем объединения содержимого одного изображения со стилем другого изображения. Целью переноса стиля является создание изображения, сохраняющего содержимое исходного изображения при применении визуального стиля другого изображения.

Формирование заголовков к изображениям — это задача описания содержания изображения словами. Эта задача лежит на стыке компьютерного

зрения и обработки естественного языка. Большинство систем, создающих подписи к изображениям используют структуру кодировщика-декодера, в которой входное изображение кодируется в промежуточное представление информации в изображении, а затем декодируется в описательную текстовую последовательность. Наиболее популярными тестами являются nocaps и COCO, а модели обычно оцениваются по метрикам BLEU или CIDER.

Локализация объекта — это задача поиска экземпляра определенной категории объектов на изображении, обычно путем указания ограничивающей рамки по центру экземпляра. В литературе задача «локализации объекта» заключается в обнаружении одного экземпляра объекта заданной категории, тогда как задача «обнаружения объекта» направлена на обнаружение всех экземпляров заданной категории на данном изображении.

Классификация сцен — это задача, в которой сцены из фотографий классифицируются по категориям. В отличие от классификации объектов, которая фокусируется на классификации заметных объектов на переднем плане, классификация сцен использует для классификации расположение объектов внутри сцены в дополнение к окружающему контексту.

Описание большинства задач компьютерного зрения с необходимыми ссылками на статьи, модели нейросетей и датасеты вы найдете на ресурсе Browse State-of-the-Art по ссылке *https://paperswithcode.com/area/computer-vision*.

Вам предлагается выбрать тему проекта с учетом указанного выше перечня (возможен выбор тематических направлений, которые не вошли в перечень). При выборе темы рекомендуется также ориентироваться на тот круг задач и возможности повторного использования кода, которые были вами изучены и разработаны в ходе выполнения лабораторных работ по ДПП ПП «Глубокие нейросети в компьютерном зрении».

2. Общее задание на проект

1. Выбрать тему проекта и согласовать её с руководителем. Подготовить проектное предложение.
2. В соответствии с темой проекта выбрать или подготовить датасет. Обеспечить загрузку данных с разбивкой их на мини пакеты. Организовать визуализацию примеров данных и провести их анализ. Обосновать и реализовать предварительную обработку данных.
3. В соответствии с решаемой задачей компьютерного зрения обосновать метод решения задачи: функцию потерь, способ регуляризации, алгоритмы оптимизации, выбор архитектуры нейросети;
4. Выполнить программирование нейросети с использованием TensorFlow/Keras/PyTorch. Проверить корректность вычислений с помощью тестов.
5. Выполнить инициализацию параметров сети, выбрать метрики для мониторинга процесса обучения сети, провести обучение сети.

6. Провести эксперименты с моделью и выполнить оценку результатов обучения нейросети. Выполнить проверку работы сети на тестовом множестве данных;

7. Оформить итоговый отчет с результатами проекта, загрузить его в ЭИОС, подготовить презентацию.

3. Календарный план проекта

Название этапа проекта	Интервал выполнения
Этап 1. Проектное предложение	15.04.2024 – 28.04.2024
Этап 2.Промежуточный отчет	29.04.2024 – 12.05.2024
Этап 3.Итоговый отчет	13.05.2024 – 26.05.2024
Этап 4.Защита проекта	27.05.2024 – 9.06.2024

4. Команда проекта

Вы можете работать в командах до 4-х человек. Мы ожидаем, что проекты, выполненные тремя-четырьмя слушателями, будут иметь более детальное описание и более весомые результаты, чем проекты, выполненные двумя слушателями.

Вы можете обращаться к любым статьям, книгам, онлайн-ссылкам или общедоступным реализациям идей и кода, которые вы, возможно, захотите включить в свой проект. Это можно делать при условии, что вы четко ссылаетесь на все источники в своем коде и своем описании проекта.

5. Этап 1. Проектное предложение

Проектное предложение должно содержать примерно 1-2 страницы текста (300-400 слов). Ваше проектное предложение должно отвечать на следующие вопросы:

Какая тема проекта?

Какой состав команды проекта (указать ФИО всех членов команды)?

Какую задачу вы собираетесь решать? Почему это интересно?

Какие статьи, материалы вы будете изучать, чтобы представить контекст и предысторию решения задачи?

Какие данные вы будете использовать? Если вам необходимо собрать новые данные, как вы это сделаете?

Какой метод или алгоритм вы предлагаете использовать? Если имеются существующие реализации, будете ли вы их использовать и как? Как вы планируете улучшать или модифицировать такие реализации? На этом этапе вам не обязательно иметь точный ответ, но вы должны иметь общее представление о том, как вы подойдете к проблеме, над которой работаете.

Каким образом вы будете оценивать результаты? Какие результаты в качественном отношении вы ожидаете представить (например, какие графики, цифровые значения)? В количественном отношении, какой анализ вы будете использовать для оценки и/или сравнения ваших результатов (например, какие метрики эффективности или статистические тесты)?

Отправка проектного предложения. Пожалуйста, прикрепите ваше проектное предложение в формате PDF в системе Мудл в разделе **Проектное предложение**. Файл с проектным предложением должен подать только один участник вашей команды. Попросите его указать ФИО остальных членов команды в имени файла с проектным предложением. Файл должен называться "*ПП_номер_команды_ФИО1_ФИО2_ФИО3_ФИО4*.pdf".

6. Этап 2. Промежуточный отчет

Промежуточный отчет о ходе проекта должен быть изложен на 2-х–3-х страницах. Отчет должен иметь следующую рекомендуемую структуру:

Название проекта, Автор(ы)

Введение: в этом разделе представляются цели и задачи проекта, общий план решения задач проекта.

Постановка задачи: содержит описание задачи, точное описание набора данных, который будет использоваться, способы оценки эффективности решения задачи и ожидаемые результаты.

Выбор и обоснование метода решения задачи: опишите и обоснуйте методы, которые вы намерены применить для решения поставленной задачи.

Полученные промежуточные/предварительные результаты: опишите и оцените результаты выполнения проекта на данный момент (наличие модуля загрузки данных, наличие модуля предобработки данных, наличие и работоспособность модуля с предварительной моделью нейросети).

Отправка промежуточного отчета: Пожалуйста, прикрепите ваш промежуточный отчет в формате PDF в системе Мудл в разделе **Промежуточный отчет.** Файл с отчетом должен подать только один участник вашей команды. Попросите его указать ФИО остальных членов команды в имени файла с промежуточным. Файл должен называться:
"*ПО_ номер_команды_ ФИО1_ФИО2_ФИО3_ФИО4*.pdf".

7. Этап 3. Итоговый отчет

Ваше окончательное описание проекта должно занимать от 8 до 12 страниц текста, структурированного по шаблону. Ниже приведена рекомендуемая структура разделов вашего отчета. Вам не обязательно

организовывать свой отчет, используя эти разделы в указанном порядке, но это будет хорошей отправной точкой для большинства проектов.

Название проекта, Автор(ы).

Аннотация: кратко опишите задачу, подход к решению задачи и ключевые результаты. Аннотация должна быть не более 300 слов.

Введение (10%): укажите цель и задачи проекта, опишите актуальность решения задач проекта, почему это важно, и кратко охарактеризуйте ожижаемые результаты.

Анализ литературы и постановка задачи (10%): проанализируйте кратко опубликованные работы, относящиеся к вашему проекту. Какие проблемы возникают при решении задачи проекта? Сформулируйте общий подход к решению задачи проекта, чем он похож на другие или чем отличается от других?

Датасет и предобработка (15%): опишите набор данных, с которым вы работаете в своем проекте. Какие типы данных образуют набор? Как вы его получили? С каким объемом данных вы работаете? Приходилось ли вам выполнять какую-либо предварительную обработку, фильтрацию или другую специальную обработку, чтобы использовать эти данные в своем проекте?

Методы (30%): опишите свой подход к решению проблем, которые вы сформулировали выше, опишите и обоснуйте методы, которые вы применили для решения задач проекта. Почему ваш подход правильный? Рассматривали ли вы альтернативные подходы? Вы должны продемонстрировать, что вы применили знания и навыки, приобретенные в ходе обучения на Программе, для решения выбранной вами проблемы. Полезно включить рисунки, диаграммы или таблицы для иллюстрации вашего метода или сравнения его с другими методами.

Оценивание результатов (30%): опишите эксперименты, которые вы провели, чтобы продемонстрировать, что выбранный вами подход решает проблему. Проводимые эксперименты могут различаться в зависимости от проекта, но можно сравнить их результаты с результатами в ранее опубликованных работах, выполнить абляционные исследования, чтобы определить влияние различных компонентов вашей нейросетевой модели на общую эффективность модели, поэкспериментировать с различными гиперпараметрами или архитектурными решениями, использовать методы визуализации, чтобы понять, как ваша модель работает, обсудите типичные режимы отказов вашей модели и т. д. Вы должны включить в отчет графики, таблицы или другие рисунки, иллюстрирующие результаты ваших экспериментов .

Заключение (5%): подведите итоги ваших основных результатов: решены ли поставленные во введении задачи? Предложите идеи для дальнейшего развития вашего проекта.

Список литературы.

Дополнительные материалы, не засчитываемые в счет в 8-12 страниц и прилагаемые в виде отдельного файла. Дополнительный материал может включать:

- исходный код (если в вашем проекте предложен алгоритм или код, который актуален и важен для вашего проекта);
- видеоролики, интерактивные визуализации, демо и т. д.

Отправка итогового отчета. Пожалуйста, прикрепите ваш итоговый отчет в формате PDF в системе Мудл в разделе **Итоговый отчет.** Дополнительные материалы прикрепите в виде отдельного PDF или ZIP файла. Файл с отчетом должен подать только один участник вашей команды. Попросите его указать ФИО остальных членов команды в имени файла с промежуточным. Файл должен называться "*ИО_ номер_команды_ ФИО1_ФИО2_ФИО3_ФИО4*.pdf". Любой код, который использовался в качестве основы проекта, должен быть указан в тексте отчета.

8. Этап 4. Защита проекта

Защита проекта будет проходить в форме презентации. Возможно также проведение стендовой сессии. Ниже приведена рекомендуемая структура презентации.

Название проекта. Автор(ы).

Постановка задачи: кратко опишите задачу, опишите общую мотивацию, а также входные и выходные данные, технические сложности, возникающие при решении задачи.

Предшествующие работы: подход и результаты.

Метод решения задачи: проиллюстрируйте используемый вами технический подход к решению задачи и инновации, приведите архитектуру модели, функцию потерь.

Результаты обучения и тестирования модели: приведите графики функции потерь, графики или значения используемых метрик, особенности выбора гиперпараметров, проведенные эксперименты и основные результаты;

Дальнейшее влияние и развитие проекта: Какое, по вашему мнению, будет влияние вашей работы? Как можно применить результаты проекта для решения сходных проблем? Каковы ограничения вашего проекта? Какие возможны направления улучшений?

Библиографический список

1. Амейзен Э. Создание приложений машинного обучения: от идеи к продукту. — СПб.: Питер, 2023. — 256 с.

2. Анирад К., Сиддха Г., Мехер К. Искусственный интеллект и компьютерное зрение. Реальные проекты на Python, Keras и TensorFlow. — СПб.: Питер, 2023 — 624 с.
3. Григорьев А. Машинное обучение. Портфолио реальных проектов. — СПб.: Питер, 2023. — 496 с.:
4. Клетте Р. Компьютерное зрение. Теория и алгоритмы / пер. с англ. А. А. Слинкин. – М.: ДМК Пресс, 2019. – 506 с.
5. Компьютерное зрение. Современные методы и перспективы развития /ред. Р. Дэвис, М. Терк; пер. с англ. В. С. Яценкова. – М.: ДМК Пресс, 2022. – 690 с.
6. Корк П. Машинное зрение. Основы и алгоритмы с примерами на Matlab / пер. с англ. В. С. Яценкова. – М.: ДМК Пресс, 2023. – 584 с.
7. Танг Д. Умные мобильные проекты с Tensorflow.- М.: ДМК Пресс, 2019. – 384 с

Приложение А. Пример проекта.

Глубокая нейросеть для классификации изображений блюд

Аннотация. Рассматривается проблема классификации изображений блюд. Продукты питания обладают уникальными характеристиками: они бывают разных цветов и форм, могут быть объединены в группы (например, фрукты, овощи и пр.) и могут комбинироваться несколькими способами для приготовления еды. Это делает изображения блюд особенно интересными для классификации. В проекте показано, что сверточные нейронные сети вполне подходят для этой задачи и превосходят традиционные подходы машинного обучения при классификации блюд.

Введение

Этот проект направлен на применение глубоких нейросетей для классификации изображений блюд. В мире существует множество кухонь; продукты питания имеют уникальный цвет, размер, форму и текстуру; их можно комбинировать несколькими способами для приготовления еды. Использование искусственного интеллекта в задаче классификации изображений блюд или продуктов питания может произвести революцию в сфере общественного питания, пропагандировать здоровое питание, сокращать пищевые отходы и др.

Мы формулируем эту задачу как задачу классификации изображения с одним классом, т.е. по изображению блюда мы хотим правильно предсказать, какое это блюдо. На рисунке 1 показаны примеры изображений двух популярных блюд. Возможность точно предсказать категорию блюда по изображению может быть полезна для нескольких сценариев применения, например, для определения количества калорий в этом продукте, определения его ингредиентов и т.д.

а) бургер б) пицца

Рисунок 1. — Пример изображений блюд

Отчет структурирован следующим образом. В разделе 1 мы рассматриваем соответствующие работы в области классификации изображений. В разделе 2 рассматриваются применяемые методы,

рейтинговая функция, функция потерь, оптимизаторы. В разделе 3 мы предоставляем подробную информацию о нашем наборе данных. В разделе 4 представлены экспериментальные результаты наших методов моделирования, а также модели нейросетей.

1. Анализ литературы и постановка задачи

Известно, что глубокие сверточные нейронные сети широко используются в задачах визуального распознавания. AlexNet [17] выиграла конкурс ImageNet по крупномасштабному визуальному распознаванию [22] в 2012 году, что вызвало большой интерес к использованию глубокого обучения для решения сложных задач. С тех пор глубокое обучение успешно используется во многих областях, таких как машинное зрение, распознавание лиц, распознавание голоса, естественное языковая обработка и т.д.

На протяжении многих лет предлагалось несколько разновидностей задач классификации изображений. Они варьируются от способности распознавать написанные цифры [18] до классификации растений по изображениям листьев [7].

Рассматриваемая нами задача классификации блюд уникальна. Ближайшим примером задачи, связанной с едой, которую мы смогли найти, была задача классификации ресторанов от Yelp [8]. В своем сценарии авторы работы [8] классифицируют изображения ресторанов по некоторым бизнес-атрибутам (например, ресторан подходит для детей, есть сервировка столиков и т.д.). Классификация блюд по изображениям имеет несколько отличительных характеристик, которые мы обсуждаем в ниже разделах 3 и 4.

2. Методы

Классификация изображений — это задача присвоения изображению (или, скорее, массиву пикселей, представляющему изображение) одной метки из фиксированного набора категорий. Полный конвейер этапов для решения этой задачи с помощью нейросети выглядит следующим образом:

- Входные данные: набор из N изображений, каждое из которых помечено одним из K различных классов. Эти данные называются обучающим набором;
- Обучение (также называемое как «Тренировка»). Использует обучающий набор, чтобы обучиться признакам каждого класса. Результатом обучения является модель, которая будет использоваться для предсказания;
- Оценивание. Оценка качества модели выполняется путем предсказания метки изображения из нового набора изображений, который она ранее «не видела» (также называемом тестовым набором). Эта оценка выполняется путем сравнения истинных меток тестового набора с предсказанными метками, формируемыми обученной моделью. Формальный подход к

решению проблемы классификации изображений можно разбить на несколько ключевых компонентов, которые мы обсудим далее.

2.1. Рейтинговая функция

Рейтинговая функция на основе данных вычисляет рейтинги классов. Для линейного классификатора рейтинговую функцию можно определить, как:

$$f(x_i, W, b) = W x_i + b. \tag{1}$$

где x_i представляет входное изображение. Матрица **W** и вектор **b** являются параметрами функции и представляют веса и смещение соответственно. При классификации изображений рейтинговая функция принимает на вход изображение x_i и вычисляет вектор $f(x_i;\mathbf{W})$ рейтингов класса (который мы обозначаем как **s**). Для входного изображения x_i предсказанный рейтинг отнесения его к j-му классу — это j-й элемент в s: $s_j = f(x_i;\mathbf{W})_j$. Далее рейтинг класса используется для вычисления потерь.

2.2 Функция потерь

Функция потерь устанавливает количественное соответствие между предсказанными и корректными метками классов обучающих данных. Функцию потерь (также называемую функцией стоимости или целевой функцией) можно рассматривать как средство оценки неудовлетворенности прогнозируемыми рейтингами, возвращаемыми рейтинговой функцией. Интуитивно понятно, что потери будут небольшими, если предсказываемые метки точно соответствуют меткам обучающих данных. В противном случае потери велики. Рассмотрим два часто используемых классификатора и их функции потерь.

2.3 Классификаторы

В этом разделе мы обсудим два распространенных классификатора, которые часто используются в задачах классификации изображений: классификатор SVM и классификатор Softmax. Для обоих из них функция, отображающая входное изображение x_i в вектор рейтингов классов $s = f(x_i;\mathbf{W})$, является одной и той же. Ниже мы рассмотрим детали каждого классификатора.

2.3.1 Классификатор SVM

Классификатор SVM использует так называемые потери максимального отступа. Для i-го входного примера данных эти потери определяются как:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta). \quad (2)$$

где Δ — гиперпараметр, который означает, что SVM функция потерь, определяемая выражением (2), «хочет» чтобы рейтинг корректного класса s_{yi} был больше, чем рейтинг неправильного класса s_j, хотя бы на величину Δ. В противном случае мы несем потери.

2.3.2. Softmax классификатор

Softmax классификатор использует кросс энтропию в качестве потерь, которая для *i*-го входного примера данных определяется в виде логарифма softmax функции:

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right) \quad \text{.....(3)}$$

Обратим внимание, что этот классификатор использует функцию softmax для отображения рейтингов класса в нормализованные положительные значения, сумма которых равна единице, так что можно эти значения рассматривать как вероятности и на их основе определять кросс энтропийные потери. Сама функция softmax определяется следующим образом

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}. \quad (4)$$

Она принимает на вход вектор действительных рейтингов (переменная *z*) и отображает их в вектор со значениями элементов в пределах от нуля до единицы, сумма которых равна единице.

2.4 Общие потери

Как для классификатора SVM, так и для классификатора Softmax полные потери на всем наборе данных равны средним потерям по всем обучающим примерам L_i плюс регуляризация *R*(W):

$$L(W) = \frac{1}{N} \sum_{i=1}^{N} L_i(x_i, y_i, W) + \lambda R(W), \quad \text{.....(5)}$$

где N представляет общее количество изображений в обучающем наборе, λ — гиперпараметр, называемый силой или коэффициентом регуляризации. Функция потерь позволяет нам количественно оценить качество любого конкретного набора параметров нашей модели: чем меньше потери, тем лучше. Рассмотрим стратегии минимизации потерь.

2.5 Оптимизация и обновление параметров

Оптимизация — это процесс поиска набора параметров модели нейросети, которые минимизируют общие потери, определенные в (5). Основной принцип методов оптимизации заключается в вычислении градиента функции потерь по отношению к параметрам модели. Градиент функции потерь определяет направление наискорейшего подъема. Одним из способов эффективного вычисления градиента является его аналитическое вычисление с использованием рекурсивного применения цепочного правила. Этот метод называется обратным распространением ошибки [19] и позволяет нам эффективно оптимизировать произвольные функции потерь. Функции потерь вычисляются с учетом различных типов сетевых архитектур (например, полносвязные нейронные сети, сверточные сети и т. д.). Обратное распространение ошибки — это основной метод вычисления градиентов для различных сетевых архитектур.

После вычисления градиента с помощью обратного распространения ошибки, градиенты используются для обновления параметров. В литературе предложено несколько подходов для выполнения обновления параметров: SGD [10], SGD+Momentum [21, 25], Nesterov Momentum [20], Adagrad [11], RMSprop [13], Adam [16] и др .

3. Датасет и предобрабока

Начнем с рассмотрения методологии сбора данных. Затем представим подробную информацию об этапах предварительной обработки данных. Завершим этот раздел описанием деталей нашего набора данных.

3.1. Множество данных

Мы подготовили наш набор данных с помощью механизмов поиска изображений Google [5] и API поиска изображений Bing [1]. Мы также исследовали использование ImageNet [6] и Flickr [3] для сбора изображений. Однако обнаружили, что изображения из Google и Bing гораздо более репрезентативны для классов, к которым они принадлежат, по сравнению с изображениями из ImageNet и Flickr. ImageNet и Flickr, похоже, содержат много поддельных изображений (изображений, которые явно не принадлежат к данному классу). Поэтому было принято решение использовать изображения, которые мы собрали с помощью Google и Bing.

3.2 Предобработка

Мы реформатировали все наши изображения, чтобы их высота, ширина и число каналов были равны 32, 32 и 3 соответственно. Это было сделано в первую очередь для повышения вычислительной эффективности при проведении наших экспериментов. Мы отфильтровали изображения, размер

которых не удалось изменить в соответствии с указанными нами требованиями по высоте, ширине и числу каналу. К сожалению, это означало потерю примерно 10% данных из нашего исходного набора данных. На рисунке 1 показаны два примера изображений из нашего набора данных. В ходе предварительной обработки осуществлялось вычитание среднего изображения из всех изображений набора данных. Среднее изображение вычислялось путем нахождения среднего арифметического пикселей изображений с использованием изображений обучающего подмножества данных.

3.3 Детали набора данных

После этапов предварительной обработки, описанных в п. 3.2, у нас было в общей сложности 26 984 изображения. Они случайным образом были разбиты на 3 непересекающихся помножества: обучающее подмножество (приблизительно 70%), валидационное подмножество (приблизительно 20%) и тестовое подмножество (приблизительно 10%). В таблице 1 указано количество изображений в каждом подмножестве. В целом набор данных содержит 20 классов. Это соответствует 20 популярным блюдам со всего мира. В таблице 2 показано распределение меток классов в наборе данных. Распределение количества изображений по классам приблизительно равномерное.

4. Оценивание результатов

Рассмотрим проведенные эксперименты и результаты. Для сравнения различных моделей мы использовали точность классификации, которую мы оценивали до двух десятичных знаков. В ходе реализации пунктов 4.1, 4.2 и 4.3 мы использовали адаптированный код из лабораторных заданий Программы “Глубокие нейросети в компьютерном зрении» [2]. В пунктах 5.4 и 5.5 мы использовали TensorFlow [9] для обучения моделей сверточных сетей.

4.1. Линейная классификации сырых данных

Чтобы установить отправную точку, мы сначала применили линейный классификатор, используя необработанные пиксели изображения в качестве признаков. Для этого мы воспользовались линейными классификаторами на основе SVM и Softmax функций. Наилучшая валидационная точность, равная 0,18 была достигнута с использованием классификатора SVM при скорости обучения 1e-07 и коэффициенте регуляризации 2,5e+04. Точность на тестовых данных при этом составила 0,16. Одной из возможных интерпретаций линейной классификации является сопоставление эталонов (шаблонов) классов, при которой каждая строка обученной матрицы весов соответствует эталонному вектору соответствующего класса. На рисунке 2 визуализированы

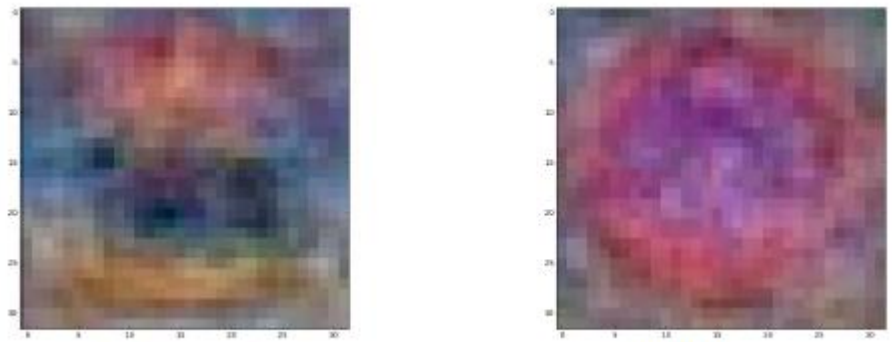
значения весов для эталонов классов гамбургер и пицца, полученные на описанном выше множестве обучающих данных. Отметим, что оба эталона соответствуют нашей интуиции: бургер содержит много коричневых пикселей, пицца имеет круглую форму и содержит много красных пикселей в центре.

Таблица 1. — Объемы данных для обучения, валидации и тестирования

Набор данных	Количество изображений
Обучающее множество	18 927
Валидационное множество	5 375
Тестовое множество	2 682

Таблица 2. — Распределение изображений по классам

Название блюда	Количество изображений
пельмени	1091
дал	1031
рамэн	1023
мороженое	1,021
наан	1020
суши	1020
кордонблю	1018
макароны	1002
лазанья 971	971
рис	966
жареная индейка	930
падтай	919
бургер	912
самса	897
буррито	888
пицца	885
колбаса	876
бирьяни	865
сэндвич	847
картофель фри	745



а) бургер б) пицца

Рисунок 2. — Визуализация весов первого слоя линейного SVM классификатора

4.2 Модели нейросетей

Следующим видом моделей, которые мы применили, были полно-связанные нейронные сети, которые также использовали необработанные пиксели изображения в качестве признаков. При этом сетевая архитектура представляла собой шестислойную полносвязную сеть. В каждом из пяти скрытых слоев было по 100 нейронов с ReLU нелинейностью, на выходе использовалась кросс-энтропийная функция потерь на основе softmax. Отметим несколько интересных наблюдений из обучения этой модели.

При обучении указанной модели оказалась полезной блочная нормализация [15]. Без блочной нормализации наша модель работала довольно плохо. Наилучшая валидационная точность, равная 0,19, была достигнута с использованием правила обновления Adam [16] при скорости обучения 1e-03. Тестовая точность составила 0,18. На рисунке 3 показана точность классификации на обучающем и валидационном множествах данных в ходе обучения на 20 эпохах.

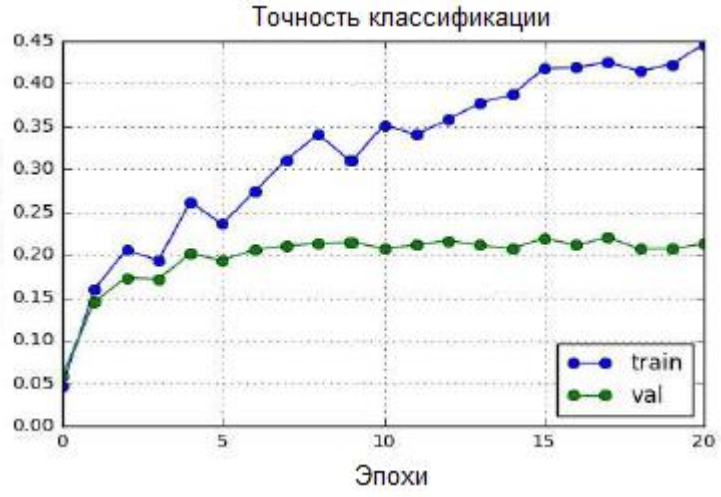


Рисунок 3. — Точность классификации для полносвязной 6-ти слойной нейронной сети

4.3 Признаки изображений

Мы также провели серию экспериментов, используя признаки, извлекаемые из изображений. Для признакового описания каждого изображения мы вычисляли гистограмму ориентированных градиентов (HOG), а также гистограмму цветов, используя канал цветов в цветовом пространстве HSV. Мы формировали окончательный вектор признаков для каждого изображения путем объединения векторов признаков HOG и гистограммы цветов. В общей сложности мы использовали 155 признаков для каждого изображения. Ниже приведены результаты классификации с предварительным вычислением указанных признаков изображений. Классификация выполнялась двухслойной полносвязной нейронной сетью с SVM функцией потерь.

При использовании признаков изображений и линейного SVM классификатора, была получена валидационнную точность, равная 0,21. Оптимизация выполнялась на основе алгоритма SGD со скоростью обучения 1e-03 и силой регуляризации 1e+00.

Использование предварительно вычисленных признаков изображений и двухслойной полносвязной нейронной сети позволило улучшить результаты. Была получена более высокая валидационная точность 0,26 при том же алгоритме оптимизации SGD со скоростью обучения 0.9, затуханием скорости обучения 0.8 и без регуляризации. Соответствующая точность на тестовом множестве составила 0,27. На рисунке 5 показана точность классификации на обучающем и валидационном множествах в течение 10 эпох при обучении 2-х слойной полносвязной нейронной сети.

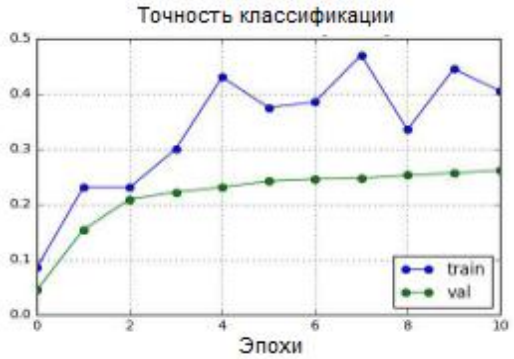


Рисунок 4. — Точность классификации 2-х слойной полносвязной нейронной сетью с использованием признаков изображения

4.4 Сверточная сеть

Для сверточной модели нейросети мы смогли получить валидационную и тестовые точности на уровне 0,4. На рисунке 4 показана использованная

нами архитектура сверточной нейросети. В ходе обучения нами были опробованы следующие практические приемы повышения эффективности процесса обучения:

- Блочная нормализация [15] оказалась весьма полезной при обучении нашей модели;
- Инициализации Хавьера [12] также оказалась полезной;
- Dropout регуляризация [14, 24] (с вероятностью сохранения 0,75) помогла повысить валидационную точность с 0,38 до 0,40;
- Использовалось фиксированное количество фильтров, равное 32;
- Использовались фильтры разных размеров (3x3, 5x5 и 7x7), но это не сильно влияло на результат, поэтому в дальнейшем использовались фильтры размером 5x5;
- Для первых трёх свёрточных слоёв мы сохраняли размеры карт активностей по высоте и ширине; Для четвертого и пятого сверточных слоев использовался maxpoooling с шагом 2;
- На выходе использовались два полносвязных слоя с 1024 и 20 нейронами соответственно;
- В качестве последнего слоя использовался softmax слой с вычислением кросс энтропийных потерь;
- Наилучшая валидационная точность, равная 0,40, была получена с использованием правила обновления Adam [16] при скорости обучения 1e-04. Точность на тестовом множестве данных составила 0,40.

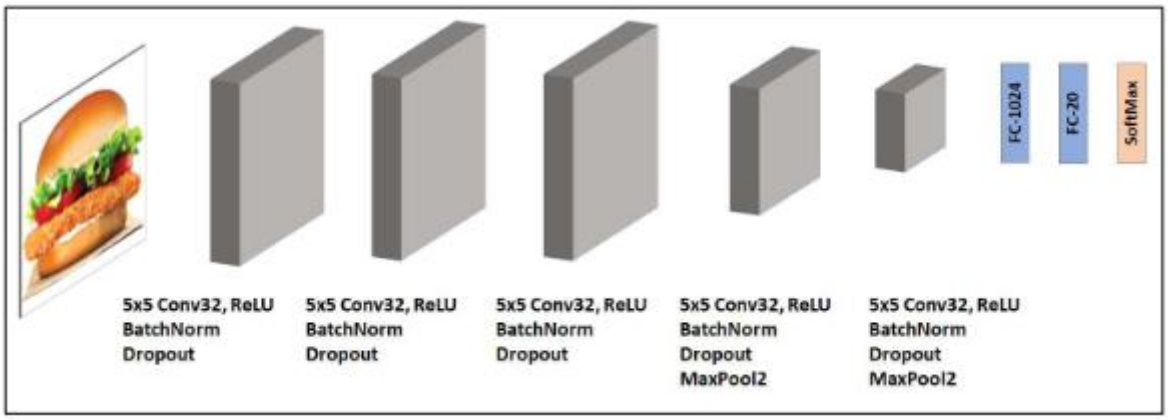


Рисунок 5. — Сверточная нейронная сеть

На рисунке 6 показано снижение потерь на нескольких итераций в первую эпоху. Мы видим, что потери вначале очень резко сокращаются, а затем постепенно выравниваются. На рисунке 7 показана точность классификации на обучающих и валидационных множествах в течение 25 эпох обучения сети.

1.2 Передача обучения

Чтобы еще больше повысить точность нашей модели, мы провели серию экспериментов по трансферному обучению. Интересно, что это дало нам

лучшие результаты на нашем наборе данных. Некоторые важные наблюдения из этого подхода заключаются в следующем:

• Использовалась модель VGG-16 [23], предварительно обученная на множестве данных ImageNet;

• Из модели VGG-16 был удален последний полносвязный слой (fc8) и заменен другим аналогичным слоем с 20 нейронами;

Процесс дообучения был организован следующим образом: сначала мы обучали последний слой в течение 10 эпох. Это позволило нам сначала получить значимые веса для слоя fc8. Затем мы дообучали всю модель на нашем наборе данных еще 10 эпох.

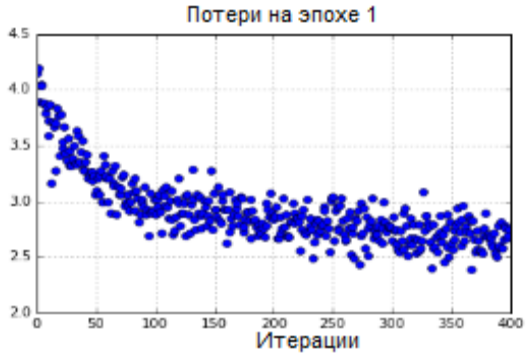


Рисунок 6. — Снижение потерь на одной эпохе для модели CNN

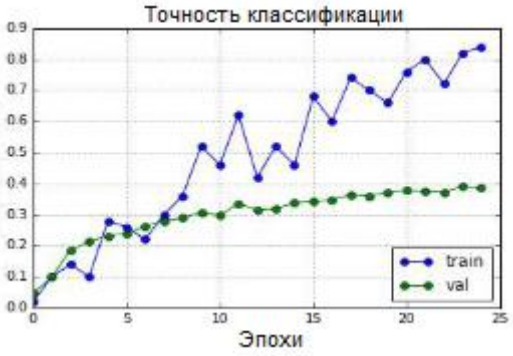


Рисунок 7. —Точность классификации для сверточной сети

Для этого подхода мы использовали пример тонкой подстройки, размещенный на GitHubGist [4]. Следуя указанному примеру, мы выполнили аналогичную предварительную обработку нашего набора данных, чтобы можно было подать его на вход модели VGG-16. Ниже перечислены этапы предварительной обработки:

• Был изменен размер изображения так, чтобы его меньшая сторона имела длину 256 пикселей. Напомним, что наш существующий набор данных имеет размеры (32, 32,3);

• Выполнялась случайная выборка кадров изображения размером 224x224 (для обучающего, валидационного и тестового множеств данных);

Осуществлялся горизонтальный поворот изображений с вероятностью 1/2 (только для обучающего множества);

• Выполнялось вычитание среднего значения каждого цвета VGG MEAN [123.68, 116.78, 103.94) (для обучающего, проверочного и тестового множеств данных).

На рисунке 8 показана точность классификации на обучающем и валидационном множествах данных в течение 20 эпох. В таблице 3 представлена сводка результатов, полученных при использовании различных моделей и практических приемов обучения.

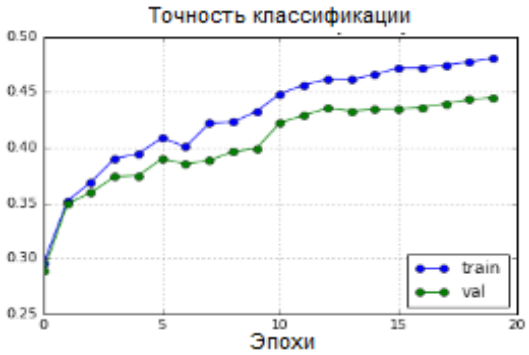


Рисунок 8. — Точность классификации после тонкой настройки модели VGG

Таблица 3. — Результаты обучения различных моделей

Модель	Валидационная точность	Тестовая точность
Линейный SVM классификатор на необработанных данных	0.18	0.16
6-ти слойная полносвязная сеть на необработанных данных	0.19	0.18
Линейный SVM классификатор на признаках изображений	0.21	0.22
2-х слойная полносвязная сеть на признаках изображений	0.26	0.27
Сверточная сеть обучаемая с нуля	0.40	0.40
Предобученнная сверточная сень (дообучение VGG модели)	0.46	0.45

Заключение

Из проведенных вычислительных экспериментов следует, что сверточные нейронные сети вполне подходят для задачи классификации блюд и превосходят в этой задаче традиционные подходы машинного обучения.

Использование переноса обучения выглядит наиболее многообещающим, особенно потому, что обучающая и валидационные точности улучшается с увеличением количества эпох (т. е. мы не переобучили

нашу модель). Это говорит о том, что увеличение объема данных (и/или их обработка в течение большего количества эпох) может улучшить показатели точности.

С точки зрения сбора данных мы планируем использовать ImageNet [6] и Flickr [3] для создания большего набора данных изображений. С точки зрения моделирования мы также планируем использовать сверточные сети в качестве средства извлечения фиксированных признаков и использовать эти признаки в ходе линейной классификации для повышения точности.

Есть несколько интересных проблем, связанных с изображениями еды, которые мы хотим исследовать в будущем. Это включает в себя возможность обнаруживать отдельные продукты на тарелке, точно предсказывать количество калорий по изображению блюда и др. Сверточные сети кажутся естественными для этих задач визуального распознавания.

Библиографический список

1. Bing image search api.
https://www.microsoft.com/en-us/bing/apis/bing-image-search-api
2. ДПП ПП Глубокие нейросети в компьютерном зрении:
https://do.sevsu.ru/course/view.php?id=10647#section-5
3. Flickr. https://www.flickr.com/
4. Moindrot O. Github. Tensorflow finetune gist
https://gist.github.com/omoindrot
5. Google image search. https://images.google.com/
6. Image-net. http://www.image-net.org/
7. Wagle S.A.; Harikrishnan R.; Ali S.H.M., Faseehuddin M. Classification of Plant Leaves Using New Compact Convolutional Neural Network Models. *Plants* **2022**, *11*, 24. https://doi.org/10.3390/plants11010024
8. Introducing the Yelp Restaurant Photo Classification Challenge: https://engineeringblog.yelp.com/2015/12/yelp-restaurant-photo-classification-kaggle.html
9. Abadi M., Barham P., Chen J., et al. Tensorflow: A system for large-scale machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA, 2016. https://arxiv.org/abs/1605.08695
10. Bottou L. Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT'2010, pages 177–186. Springer, 2010.
https://leon.bottou.org/publications/pdf/compstat-2010.pdf
11. Duchi J., Hazan E., and Singer Y.. Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12(Jul):2121–2159, 2011. https://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf
12. Glorot X.and Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In Aistats, volume 9, pages 249–256, 2010. https://proceedings.mlr.press/v9/glorot10a/glorot10a.pdf
13. Hinton G., Srivastava N. and Swersky K. Neural networks for machine learning lecture 6a overview of mini–batch gradient descent. 2012.
https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf
14. Hinton G. E., Srivastava N., Krizhevsky A., et al Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580, 2012.
15. Ioffe S. and Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015.

16. Kingma D. and Ba J.. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

17. Krizhevsky A., Sutskever I., and Hinton G. E. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.
https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

18. LeCun Y., Boser B. E., Denker J. S., et al. Handwritten digit recognition with a back-propagation network. In Advances in neural information processing systems, pages 396–404, 1990.
https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf

19. LeCun Y. A., Bottou L., Orr G. B., and M¨uller K.-R. Efficient backprop. In Neural networks: Tricks of the trade, pages 9–48. Springer, 2012.
https://link.springer.com/chapter/10.1007/978-3-642-35289-8_3

20. Нестеров Ю.Е. Метод решения задачи выпуклого программирования со скоростью сходимости $O(1/k^2)$. Доклады АН СССР, том 269, с. 543-547,1983.
https://www.mathnet.ru/links/de4babb15e427718d961a24f7451afad/dan46009.pdf

21. Qian N. On the momentum term in gradient descent learning algorithms. Neural networks, 12(1):145–151, 1999.
https://www.sciencedirect.com/science/article/pii/S0893608098001166

22. Russakovsky O., Deng J., Su H., et al. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211–252, 2015. https://arxiv.org/abs/1409.0575

23. Simonyan K. and Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

24. Srivastava N., Hinton G., Krizhevsky A., et al. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1):1929–1958, 2014.
https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

25. Sutskever I., Martens J., Dahl G., and Hinton G. On the importance of initialization and momentum in deep learning. In International conference on machine learning, pages 1139–1147, 2013. http://proceedings.mlr.press/v28/sutskever13.pdf