ПРАКТИЧЕСКАЯ РАБОТА №4: НОРМАЛИЗАЦИЯ ДАННЫХ И ИССЛЕДОВАНИЕ РАЗЛИЧНЫХ АРХИТЕКТУР НЕЙРОННЫХ СЕТЕЙ

Цель

Изучить основные методы нормализации и регуляризации в нейронных сетях, исследовать их влияние на процесс обучения и качество модели. Провести эксперименты с различными архитектурами нейронных сетей (ResNet, VGG, MobileNet) и сравнить их эффективность.

Задание

Часть 1: Нормализация данных (1 пара)

1. Подготовка данных:

• Используйте датасет CIFAR-10 или другой из OpenML/Kaggle.

• Выполните предварительную обработку: нормализация пиксельных значений изображений.

2. Применение методов нормализации:

• Постройте полносвязную или сверточную нейронную сеть и обучите её без нормализации данных.

• Добавьте следующие методы нормализации и повторите обучение:

• Batch Normalization

• Layer Normalization

• Dropout

3. Анализ:

• Сравните результаты обучения при использовании разных методов нормализации.

• Оцените уровень переобучения и скорость сходимости для каждого метода.

4. Визуализация:

• Постройте графики потерь (loss) и точности для всех методов нормализации.

Часть 2: Исследование различных архитектур нейронных сетей (1 пара)

1. Выбор архитектур:

• Обучите три разные архитектуры сверточных сетей на одном и том же наборе данных:

• ResNet (Residual Networks)

• VGG (Very Deep Convolutional Networks)

• MobileNet (для мобильных и маломощных устройств)

2. Эксперименты:

• Проведите обучение для 20 эпох, используя одинаковые гиперпараметры (learning rate, batch size).

• Визуализируйте процесс обучения через TensorBoard или Matplotlib.

3. Сравнительный анализ:

• Сравните точность на тестовой выборке, количество параметров и время обучения каждой архитектуры.

• Проанализируйте, какая архитектура лучше справляется с переобучением.

Что нужно вставить в отчет:

1. Введение:

• Цели работы и краткое описание нормализации и различных архитектур.

• Описание используемого датасета.

2. Эксперименты с нормализацией:

• Схема нейронной сети, на которой проводился эксперимент.

• Графики потерь и точности для различных методов нормализации.

• Таблица с основными метриками (точность, скорость сходимости).

• Сравнительный анализ: как нормализация влияет на эффективность модели и переобучение.

3. Сравнительный анализ архитектур:

• Описание архитектур ResNet, VGG и MobileNet.

• Графики потерь и точности на обучающей и тестовой выборках.

• Таблица с параметрами (размер модели, время обучения, точность).

4. Выводы:

• Какая архитектура показала лучшие результаты для данного набора данных.

• Влияние нормализации на процесс обучения и переобучение.

• Рекомендации по применению различных архитектур в зависимости от задачи.

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ ПО 4 ПРАКТИЧЕСКОЙ

Шаг 1. Формулировка цели и общая мотивация

1.1. Необходимость нормализации и регуляризации

Нормализация

При обучении глубокой нейронной сети параметры (веса и смещения) подбираются методом стохастического градиентного спуска (SGD) или его вариаций (Adam, RMSProp и т. д.).

Скорость и стабильность обновления весов напрямую зависят от масштаба и распределения входных данных. Если признаки (например, пиксели изображений) сильно различаются по порядку величины, процесс оптимизации затрудняется.

Нормализация входных данных (приведение признаков к единому масштабу) способствует упрощению ландшафта оптимизации и позволяет быстрее достичь оптимума.

Регуляризация

В ходе обучения сеть может переобучиться, «запоминая» паттерны из обучающей выборки и теряя способность к обобщению.

Методами борьбы с переобучением (регуляризации) являются L1/L2-регуляризация, Dropout, нормализующие слои (BN, LN) и др.

Корректно подобранная регуляризация повышает качество модели на тестовых данных и делает обучение более устойчивым к шуму.

Практическая задача

Целью практической работы является исследование различных методов нормализации и сравнение трёх популярных архитектур сверточных сетей

(ResNet, VGG, MobileNet).

Итогом должно стать понимание, как сказывается выбор метода нормализации и архитектуры на качество модели (точность, переобучение, время обучения, число параметров).

Шаг 2. Нормализация данных: детальный обзор

2.1. Нормализация входных признаков

Min-Max нормализация

Приводит каждый признак (пиксель изображения) к диапазону [0, 1].

Формула:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Применяется часто в компьютерном зрении для преобразования изображений, когда исходные пиксели имеют диапазон [0, 255].

Z-преобразование (StandardScaler)

Вычитает среднее и делит на стандартное отклонение.

Формула:

$$x_{norm} = \frac{x - \mu}{\sigma}$$

Удобна при работе со скалярными или векторными признаками, где важно привести все признаки к «центру» с нулевым средним и единичным стандартным отклонением.

Дополнительные замечания

Размерность и тип данных влияют на выбор конкретного типа нормализации. В компьютерном зрении нередко делают глобальное усреднение (по всем изображениям) или по каналам (R, G, B).

Важно корректно сохранять статистики обучающей выборки при

тестировании, чтобы не вносить смещений.

Шаг 3. Методы нормализации внутри нейронной сети

3.1. Batch Normalization (BN)

Алгоритмические основы

Для каждого обучающего мини-батча (batch), содержащего B объектов, вычисляются среднее \mu и дисперсия \sigma^2 активаций (например, выходов сверточного слоя).

Затем активации нормализуются по формуле:

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

где ϵ — небольшая добавка для численной стабильности.

После этого применяется обучаемая линейная трансформация с двумя параметрами γ и β:

$$y = \gamma \times \hat{x} + \beta$$

Преимущества

Ускоряет сходимость, так как снижает проблему «дрейфа» активаций внутри сети (internal covariate shift).

Дает частичный эффект регуляризации, поскольку использует статистику конкретного батча, а не всего датасета.

Уменьшает чувствительность к выбору начальных весов и learning rate.

Практические аспекты

BN обычно вставляют сразу после линейного или сверточного слоя и до функции активации (хотя встречаются варианты размещения).

На этапе инференса (или валидации) используются скользящие усреднённые статистики, накопленные за время тренировки.

3.2. Layer Normalization (LN)

Принцип работы

В отличие от BN, LN нормирует активации по всем признакам (нейронам) в пределах одного объекта, а не по выборке (батчу).

Для входного вектора вычисляется среднее и дисперсия по всем компонентам этого вектора.

Когда использовать

Удобна при небольших размерах батча, где статистика BN может быть неточной.

Хорошо подходит для рекуррентных нейронных сетей и трансформеров, где обрабатываются последовательности.

3.3. Dropout

Механизм работы

На каждом шаге обучения случайно зануляется (соответствующий выход становится равным нулю) фиксированный процент нейронов слоя.

Отключение нейронов создает эффект обучения «нескольких усреднённых сетей», повышая способность обобщать.

При тестировании все нейроны включены, но эффекты случайного отключения компенсируются корректировкой (масштабированием) весов.

Особенности применения

Чаще всего слой Dropout размещают перед выходным полносвязным слоем или между сверточными блоками.

Параметр p (вероятность зануления) обычно выбирается из диапазона 0.2–0.5. Если p слишком велик, сеть может недоучиваться; если слишком мал, переобучение незначительно снижается.

Шаг 4. Архитектуры сверточных нейронных сетей: подробное описание

В данной работе предполагается эксперимент с тремя широко применяемыми архитектурами: ResNet, VGG и MobileNet. Каждая из них имеет свои особенности в отношении глубины, структуры свёрток и оптимизации.

4.1. ResNet (Residual Networks)

Идея остаточных связей

Традиционная проблема очень глубоких сетей — затухание градиента, затрудняющее обучение.

ResNet решает это за счёт блоков, где выходные данные сверточных слоёв складываются с «проброшенными» (skip connection) входными данными.

Такой дизайн позволяет градиенту беспрепятственно распространяться назад, упрощая оптимизацию.

Особенности реализации

ResNet-18, ResNet-34, ResNet-50 и т. д. отличаются количеством слоёв и типов блоков (BasicBlock для меньшей глубины, BottleneckBlock для большей).

Существует множество модификаций (ResNeXt, Wide ResNet и прочие), расширяющих оригинальную идею.

4.2. VGG (Very Deep Convolutional Networks)

Основная концепция

Сеть строится из нескольких «блоков», где каждый блок содержит несколько сверточных слоёв 3×3 подряд, за которыми следует слой подвыборки (Pooling).

Глубина сети увеличивается постепенным добавлением сверточных слоёв в каждом блоке (VGG16, VGG19 и т. д.).

Плюсы и минусы

Простая архитектура, легко воспроизводимая.

Большое количество параметров (особенно в полносвязных слоях), что увеличивает требования к памяти и вычислительной мощности.

4.3. MobileNet

Depthwise separable convolutions

Основной принцип — разделить обычную сверточную операцию (когда фильтр свёртывается по всем каналам и пространству одновременно) на две части:

Depthwise Convolution: свёртка по отдельным каналам.

Pointwise Convolution (1×1): «смешивание» каналов после depthwise.

Такой подход позволяет сократить количество умножений и параметров.

Применение

MobileNet, MobileNetV2, MobileNetV3 ориентированы на работу в условиях ограниченных ресурсов (смартфоны, микроконтроллеры).

Несмотря на компромисс в точности по сравнению с «большими» сетями, MobileNet обеспечивает высокую скорость работы и малый размер модели.

Шаг 5. Практическое руководство по проведению экспериментов

Данный раздел содержит детализированный план выполнения практической работы.

5.1. Подготовка данных

Выбор датасета

Рекомендуется датасет CIFAR-10 (10 классов, 50k тренировочных изображений, 10k тестовых).

Можно использовать альтернативы (OpenML, Kaggle), но следует убедиться в совместимом формате (например, изображения 32×32 или 64×64).

Предварительная обработка

Выполните загрузку датасета (через фреймворк PyTorch, TensorFlow или Keras).

Примените Min-Max или Z-преобразование к пикселям.

Разделите данные на обучающую и тестовую выборки (часто дополнительно выделяют валидационную выборку).

5.2. Исследование методов нормализации

Базовая модель без нормализации

Создайте простую модель (например, небольшая сверточная сеть с 2–3 сверточными слоями и 1–2 полносвязными слоями) или полносвязную сеть для более простых задач.

Обучите эту модель на 5–10 эпох, сохраните кривые loss/accuracy для анализа.

Batch Normalization

Добавьте слои BN после каждого сверточного (или линейного) слоя.

Обучите модель столько же эпох, зафиксировав остальные гиперпараметры (learning rate, batch size).

Сравните полученные кривые loss/accuracy со случаем без BN.

Layer Normalization

Замените BN на LN в соответствующих местах.

Повторите процесс обучения, визуализируйте динамику loss/accuracy.

Обратите внимание на эффект LN при малом размере batch.

Dropout

Добавьте Dropout (например, p=0.5) перед заключительными полносвязными слоями или в середине сети.

Оцените, насколько упадёт переобучение (разница между accuracy на трейне и тесте).

Сопоставьте графики с предыдущими результатами.

5.3. Сравнительный анализ архитектур (ResNet, VGG, MobileNet)

Подготовка моделей

Используйте готовые реализации архитектур из стандартных библиотек (PyTorch torchvision.models, TensorFlow Keras Applications и т. д.).

Для CIFAR-10 обычно подходят «урезанные» версии ResNet, VGG, MobileNet (например, ResNet18, VGG11, MobileNetV2).

Единые гиперпараметры

Batch size: 64 или 128 (в зависимости от доступной памяти).

Learning rate: 0.001–0.01 (подбирается эмпирически).

Число эпох: не менее 20, чтобы увидеть устойчивую динамику обучения.

Сбор данных

Фиксируйте время обучения каждой эпохи.

После каждой эпохи замеряйте точность (accuracy) на валидационной/тестовой выборке.

Сохраните итоговое число параметров (можно получить через средства фреймворка — например, model.summary() в Keras).

Анализ результатов

Постройте графики loss/accuracy.

Создайте таблицу, в которую внесите следующие параметры:

Итоговая точность на тесте (лучшее значение за все эпохи).

Время обучения 1 эпохи (или суммарное время за все эпохи).

Число параметров сети (порядок миллионов).

Субъективная оценка склонности к переобучению (сравнение кривых тренировочной и тестовой точности).

Шаг 6. Подготовка отчёта и формирование выводов

6.1. Структура отчёта

Введение

Сформулируйте цели практической работы, объясните актуальность исследования нормализации и различных сетевых архитектур.

Кратко опишите датасет (количество классов, объём обучающей выборки).

Теоретический обзор

Кратко изложите суть методов нормализации (BN, LN, Dropout), обоснуйте их применение.

Опишите основные идеи и особенности архитектур ResNet, VGG, MobileNet.

Экспериментальная часть

Детально опишите конфигурацию использованной (базовой) нейронной сети для экспериментов с нормализацией.

Представьте результаты (графики loss/accuracy, таблицы метрик) для разных методов нормализации:

Без нормализации.

С Batch Normalization.

С Layer Normalization.

С Dropout.

Изложите сравнительный анализ, отметьте, какой метод или комбинация методов приводит к лучшим показателям.

Сравнение архитектур

Опишите процедуру обучения ResNet, VGG и MobileNet (число эпох, размер батча, функция потерь).

Сформируйте таблицы, показывающие итоговую точность, время обучения и число параметров.

Проанализируйте, какая архитектура лучше справляется с ограниченными ресурсами, какая достигает более высокой точности, как быстро сходится обучение.

Выводы

Сформулируйте, какие методы нормализации наиболее эффективны для данного датасета в условиях вашей экспериментальной среды.

Отметьте, при каких условиях (размер batch, глубина сети) BN, LN или Dropout дают наибольший эффект.

Укажите наиболее подходящую архитектуру (или баланс архитектуры и нормализации) для задач, похожих на CIFAR-10 (компьютерное зрение, ограниченные ресурсы и т. д.).

6.2. Рекомендации и перспективы

Выбор метода нормализации

Если размер батча достаточно велик, предпочтительнее Batch Normalization (ускоряет и стабилизирует обучение).

При работе с последовательностями или очень малыми батчами целесообразно заменить BN на Layer Normalization.

Dropout полезен, когда данные относительно невелики, а сеть склонна к переобучению.

Выбор архитектуры

ResNet: оптимальный вариант при необходимости строить глубокую модель (например, ResNet50, ResNet101) и при наличии достаточной вычислительной мощности.

VGG: проще в понимании, но требует больше памяти; подходит для исследовательских задач, где удобство имеет приоритет над оптимальностью.

MobileNet: лучшее решение для мобильных и встраиваемых систем, где критичны как размер модели, так и время ответа.

Дальнейшие исследования

Изучение комбинированных методов (например, одновременное применение BN и Dropout).

Эксперименты с разными политиками изменения learning rate (LR scheduling).

Пробное применение различных оптимизаторов (SGD, Adam, RMSProp) в сочетании с нормализацией, чтобы найти наилучшие связки.