МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ КЫРГЫЗСКОЙ РЕСПУБЛИКИ

КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И. РАЗЗАКОВА

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

СОЗДАНИЕ МОДЕЛИ ДЛЯ КЛАССИФИКАЦИИ AI-СГЕНЕРИРОВАННЫХ И РЕАЛЬНЫХ ИЗОБРАЖЕНИЙ

Выполнили: ст. гр. ПМИ-3-22
Ишенов Д. Т.

Проверила: ст. преп. Токтоналы А.

Бишкек – 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные технологии искусственного интеллекта (ИИ) находят широкое применение в различных областях, включая обработку и генерацию изображений. С развитием генеративных моделей, таких как GAN (Generative Adversarial Networks), стало возможно создавать изображения, которые практически невозможно отличить от реальных. Однако это создает новые вызовы, связанные с необходимостью распознавания искусственно сгенерированных изображений, особенно в контексте проверки подлинности контента и предотвращения распространения дезинформации.

Целью данной работы является разработка веб-приложения, способного классифицировать изображения как реальные или сгенерированные с использованием технологий ИИ. В основе приложения лежит модель глубокого обучения, обученная на реальных и сгенерированных данных. Использование веб-фреймворка Django позволяет интегрировать предсказания модели в удобный пользовательский интерфейс, что делает данное решение доступным для широкой аудитории.

В реферате описаны основные этапы работы: выбор и обучение модели нейронной сети, разработка веб-приложения для обработки изображений и визуализации результатов, а также интеграция модели в серверное приложение. Реализация проекта демонстрирует возможности использования современных технологий глубокого обучения и веб-разработки для решения актуальных задач классификации изображений.

ГЛАВА 1. ТЕХНОЛОГИИ И ПОДХОДЫ В КОМПЬЮТЕРНОМ ЗРЕНИИ

Классификация изображений – это задача машинного обучения, в рамках которой алгоритм должен анализировать изображение и определить, к какой категории оно относится. В отличие от задач, где входные данные представляют собой текстовую или числовую информацию, изображения являются высокоразмерными данными с множеством особенностей, которые необходимо извлечь для принятия правильного решения.

1.1. Основы классификации изображений

Как же работает классификация? Когда ваша система получает входное изображение, ей уже известен фиксированный набор категорий или меток. Это могут быть любые объекты: «кошка», «собака», «самолёт», «грузовик» и так далее. Компьютер должен посмотреть на изображение и назначить ему одну из меток.[1]

Со стороны задача выглядит несложной, поскольку большая часть нашей зрительной системы запрограммирована на распознавание объектов. Но для машины это не так-то просто.

Машинное обучение для классификации изображений обычно состоит из двух этапов: обучения и предсказания.

Этап обучения:

На этом этапе алгоритм машинного обучения получает большое количество размеченных данных — в данном случае изображения, на которых указана их категория (например, «собака» или «кошка»). Модель обучается на этих данных, извлекая из них признаки, которые наиболее

важны для правильной классификации. Один из самых популярных подходов для классификации изображений — это использование глубоких нейронных сетей, таких как сверточные нейронные сети (CNN), которые обладают способностью автоматически обучаться на изображениях.

Этап предсказания:

После обучения модель может предсказывать категорию для новых, ранее не виденных изображений. В этом случае она будет использовать те признаки, которые она извлекла в процессе обучения, чтобы классифицировать новое изображение в одну из категорий. Модель будет работать с изображением, преобразуя его в числовое представление и проходя через слои нейронной сети, пока не сделает предсказание.

Основные типы моделей для классификации изображений

Существует несколько типов моделей, которые широко применяются для классификации изображений. Некоторые из них включают:

1. Логистическая регрессия

Это один из самых простых методов, который подходит для задач классификации с небольшим количеством признаков. Однако в контексте изображений логистическая регрессия редко используется, поскольку изображения содержат слишком много информации, и такой подход не может эффективно работать с ними.

2. Сверточные нейронные сети (CNN):

Это один из наиболее эффективных методов для классификации изображений. CNN использует специализированные слои (свертки), которые

помогают модели выявить локальные зависимости в изображениях. С помощью этих слоев сеть может распознавать такие элементы, как края, текстуры, формы и другие важные признаки, что делает её идеальной для задач компьютерного зрения.

3. Рекуррентные нейронные сети (RNN):

Хотя RNN традиционно применяются для обработки последовательных данных (например, текста), их можно использовать и для классификации изображений, если изображения интерпретировать как последовательности пикселей. Однако RNN не так часто используется в задачах классификации изображений по сравнению с CNN, поскольку они менее эффективны для этой цели.

4. Сети с остаточными связями (ResNet):

Это глубокая нейронная сеть, разработанная для решения проблемы затухания градиента. Она использует концепцию «skip connections» или «residual connections», позволяющих передавать информацию непосредственно от одного слоя к другому, минуя промежуточные слои. Это позволяет обучать более глубокие сети с лучшей производительностью.

1.2. Использование сверточных нейронных сетей

Сверточная нейронная сеть (англ. convolutional neural network, CNN) — специальная архитектура нейронных сетей, предложенная Яном Лекуном, изначально нацеленная на эффективное распознавание изображений[2].

CNN имитируют работу человеческого зрения, выделяя характерные особенности изображения через серию преобразований. Основная идея заключается в том, чтобы использовать свёртки (convolutions) для извлечения признаков из входных данных.

Основные компоненты CNN:

1. **Свёрточный слой (Convolutional Layer):**

   В этом слое применяется операция свёртки, которая вычисляется как:

$$Z_{i,j}^{(k)} = \sum_m \sum_n X_{i+m,j+n} \cdot W_{m,n}^{(k)} + b^{(k)}$$

где:

- X *– входное изображение,*
- $W^{(k)}$ *– фильтр (ядро свёртки),*
- $b^{(k)}$ *– смещение (bias),*
- $Z^{(k)}$ *– результат применения фильтра.*

Этот процесс позволяет выделить локальные особенности, такие как края, углы или текстуры.

2. **Слой активации (Activation Layer):**

   После применения свёртки результат проходит через нелинейную функцию, например, ReLU (Rectified Linear Unit):

$$f(x) = max(0, x)$$

ReLU добавляет нелинейность, что позволяет модели лучше представлять сложные зависимости в данных.

3. **Слой подвыборки (Pooling Layer):**

   Для уменьшения размерности данных используется подвыборка, обычно max pooling:

$$Z_{i,j} = \max(X_{i:i+p,j:j+q})$$

Это помогает сократить вычислительные затраты и сделать модель более устойчивой к шуму.

4. **Полносвязный слой (Fully Connected Layer):**

   После нескольких свёрточных и подвыборочных слоёв данные передаются в полносвязный слой, который выполняет классификацию.

Архитектура модели

Для проекта классификации изображений мы используем простую архитектуру CNN:

1. **Входные данные:** изображения размером 300×300×3
2. **Первый свёрточный блок:**
   - Свёртка с 32 фильтрами (3×3), активация ReLU.
   - Подвыборка (max pooling) размером 2×2.
3. **Второй свёрточный блок:**
   - Свёртка с 64 фильтрами (3×3), активация ReLU.
   - Подвыборка (max pooling).
4. **Третий свёрточный блок:**

- Свёртка с 128 фильтрами (3×3), активация ReLU.
- Подвыборка (max pooling).

5. **Полносвязный слой:**
   - Уплощение данных (flattening).
   - Полносвязный слой с 128 нейронами, активация ReLU.
6. **Выходной слой:**
   - Один нейрон с активацией sigmoid для предсказания вероятности класса (AI-Generated или Real).

Реализация в проекте

Для классификации изображений в проекте была обучена модель CNN. Она обучалась на двух классах: изображения, сгенерированные искусственным интеллектом, и реальные фотографии. Использовался оптимизатор Adam, а функция потерь — Binary Crossentropy:

$$\text{Binary Crossentropy} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right)$$

где y_i – истинная метка, а y^*_i – предсказанная вероятность.

На этапе предсказания изображение проходит следующие этапы:

1. Преобразование к нужному размеру и нормализация.
2. Пропуск через модель для получения вероятности.
3. Классификация на основе порога вероятности (например, 0.5)

ГЛАВА 2. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ ПРОЕКТА

2.1 Разработка модели классификации изображений

На этапе разработки модели классификации изображений основное внимание было уделено выбору архитектуры, предобработке данных и процессу обучения. Данные были взяты с платформы Hugging Face[3].

1. **Предобработка данных:**
   Данные разделены на два класса: изображения, созданные искусственным интеллектом, и реальные фотографии. На этапе загрузки:
   - Изображения приводились к размеру 300×300×3.
   - Пиксели нормализованы в диапазон [0, 1].
2. **Архитектура модели:**
   Использована сверточная нейронная сеть (CNN), адаптированная для двух классов. Архитектура включает:
   - Три сверточных блока (свёртка, активация ReLU, max pooling).
   - Полносвязный слой для объединения признаков.
   - Выходной слой с одной нейронной сигмоидальной активацией, предсказывающий вероятность принадлежности к классу.
3. **Обучение модели:**
   - Оптимизатор: Adam, известный своей быстрой сходимостью.
   - Функция потерь: Binary Crossentropy, подходящая для задач бинарной классификации.
   - Метрики: Точность (accuracy) и потери (loss) отслеживались на обучающем и валидационном наборах.

4. **Существующие проблемы:**
    - Переобучение: На данный момент существует такая проблема как переобучение, то есть, негативное явление, возникающее, когда алгоритм обучения вырабатывает предсказания, которые слишком близко или точно соответствуют конкретному набору данных и поэтому не подходят для применения алгоритма к дополнительным данным или будущим наблюдениям. Сейчас моя модель может и хорошо справляется с тестовыми данными, но с ввалидационными данными как мы видим справляется гораздо хуже, даже со временем результаты ухудшается. Я пытался добавлять *dropout* для предотвращения этого, но результаты не улучшились. Мне еще предстоит над этим работать.

2.2 Интеграция модели с веб-приложением

Для практического использования разработанной модели она была интегрирована в веб-приложение, позволяющее загружать изображения и получать предсказания в реальном времени.

1. **Инструменты и фреймворки:**
    - **Django:** Это свободный фреймворк для разработки быстрых и безопасных веб-приложений и сайтов на языке Python. Использует шаблон проектирования MVC..
    - **PyTorch:** Использован для загрузки обученной модели.
2. **Архитектура веб-приложения:**
    - **Frontend:** HTML-форма для загрузки изображений, JavaScript для отправки запросов и отображения результатов.

- **Backend:** В Django настроен эндпоинт для обработки запросов. Алгоритм:
  - Получить загруженное изображение.
  - Преобразовать его в тензор.
  - Передать тензор в модель и получить предсказание.
  - Вернуть ответ в формате JSON (предсказанный класс и вероятность).

3. **Проблемы и их решение:**
   - **Совместимость форматов:** Использованы библиотеки Pillow и torchvision для преобразования изображений.
   - **Производительность:** Для обработки изображений и работы модели использован MacBook Air с чипом M1.

Рис. 1. Веб-приложение AI Image Classifier

2.3 Оценка результатов и выводы

Оценка точности:
Модель показала среднюю точность (около 62%) на валидационных данных и около 94% на тестовых данных, что является явным признаком переобучения.

Результаты модели:

- **Точность (Accuracy):** 62.2%
- **Точность предсказаний (Precision):** 62.2%
- **Полнота (Recall):** 100%
- **F1-мера:** 71.13%

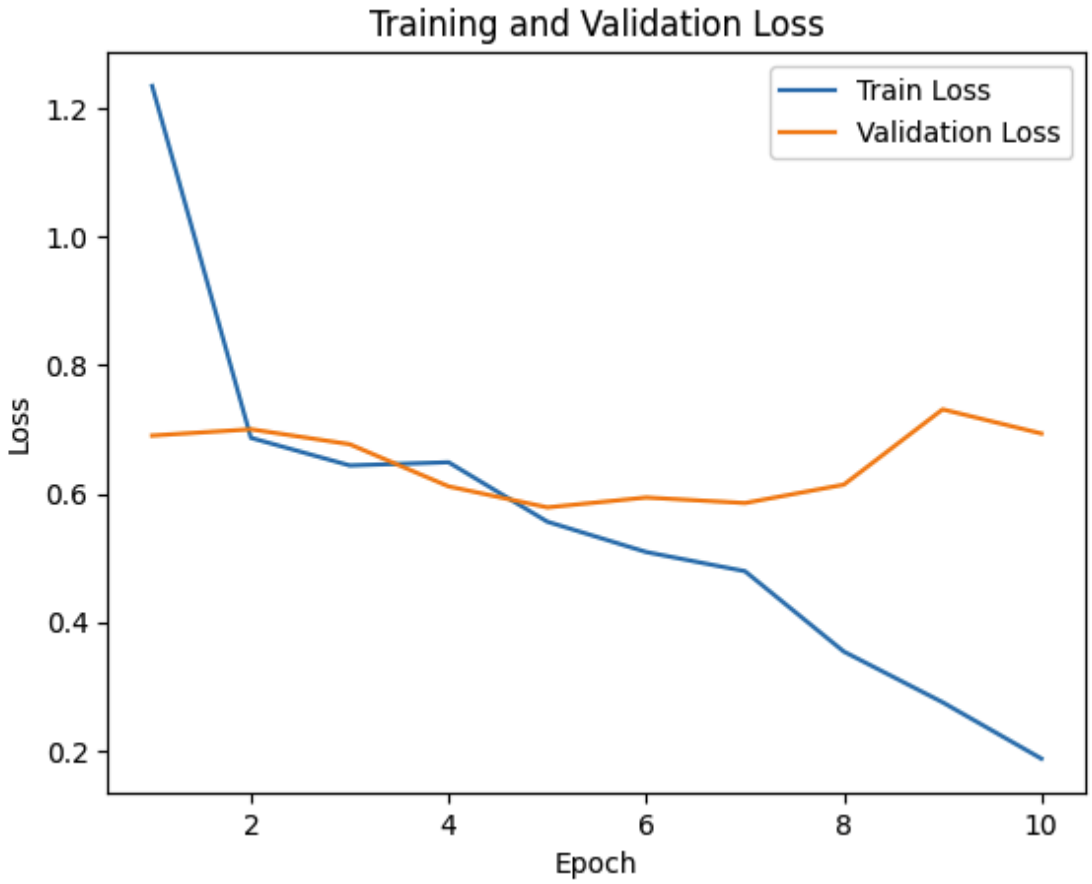

Рис. 2. Training and Validation Loss

ЗАКЛЮЧЕНИЕ

В процессе работы над проектом я разработал модель для классификации изображений с использованием методов машинного обучения, а именно сверточных нейронных сетей (CNN). Модель была интегрирована в веб-приложение, которое позволяет пользователям загружать изображения и получать предсказания, определяя, является ли изображение сгенерированным искусственным интеллектом или настоящим. Однако, несмотря на успешную разработку, результаты показали необходимость дальнейших улучшений, так как точность модели на новых данных оказалась ниже ожидаемой.

При обучении модели использовался набор данных, состоящий из изображений двух категорий: «AI-Generated» и «Real». Модель показала удовлетворительные результаты на обучающих данных, но ее точность на новых изображениях с другого источника была ниже, что указывает на переобучение модели. В итоге точность модели составила 62%, что является ниже желаемого показателя.

Разработка веб-приложения позволила мне интегрировать модель с пользовательским интерфейсом, обеспечив удобство работы с системой. Пользователи могут загружать изображения, а система возвращает результат предсказания с вероятностью. Веб-приложение продемонстрировало свою работоспособность, несмотря на невысокую точность модели, и обеспечивало быстрый отклик при обработке запросов.

Разработанная система классификации изображений с использованием сверточных нейронных сетей и интеграция модели в веб-приложение показали свою функциональность, но для повышения ее точности и универсальности необходимы дальнейшие доработки. В будущем я планирую

значительно улучшить проект с использованием более мощных архитектур и методов регуляризации, что позволит достичь более высоких результатов.

СПИСОК ИСТОЧНИКОВ

1. Стэнфордский курс: лекция 2. Классификация изображений *[https://www.reg.ru/blog/stehnfordskij-kurs-lekciya-2-klassifikaciya-izobrazhenij]*
2. Сверточные нейронные сети *[https://neerc.ifmo.ru/wiki/]*
3. Hemg/AI-Generated-vs-Real-Images-Datasets *[https://huggingface.co/datasets/Hemg/AI-Generated-vs-Real-Images-Datasets]*
4. **PyTorch Documentation.** *Transfer Learning for Computer Vision Tutorial* Официальная документация по использованию PyTorch для классификации изображений.