**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

*техника*

*Профиль:**Вычислительные системы и сети*

Выполнили:  
студенты группы УВВ-211

Демчук И.А.

Ахтырский М.С.

Проверил:

Ст.преп. Абрамов А.В.

Доцент Малинский С.В.

(должность, ФИО)

**МОСКВА 2023**

**Общее задание на проектную деятельность на 3 семестр для студентов группы УВВ-211**

Для каждого варианта заданы:

- сформировать не менее 300 изображений (общую выборку) согласно теме задания;

- из общей выборки сформировать обучающую выборку, размерностью порядка 230-250 объектов;

- сформировать контрольную выборку для оценки качества распознавания;

- разделить на классы;

- выбрать модель обучения и провести машинное обучение (программно реализовать систему распознавания) с контролем качества обучения;

- реализовать систему распознавания в виде программы с графическим интерфейсом;

- продемонстрировать результат на контрольной выборке визуально и с оценкой качества.

**АННОТАЦИЯ**

Пояснительная записка, 81с., 7рис., 13 таблиц, 3 приложения

Объектом исследования является процесс проектирования систем искусственного интеллекта с использованием открытых программ для решения задач машинного обучения.

Предмет исследования – методы и техники машинного обучения, доступные через открытые программы.

Целью исследования является разработка системы, использующей открытые программы для решения задач машинного обучения.

В ходе работы над проектом был проведен анализ существующих открытых программ для машинного обучения, установлены задачи и области применения системы. Также был проведен обзор методов машинного обучения, из которых были выбраны наиболее подходящие для нашего проекта.

**ВВЕДЕНИЕ**

В современном мире технологии искусственного интеллекта (ИИ) играют все более значимую роль в различных сферах человеческой деятельности. Применение открытых программ для решения задач машинного обучения становится ключевым элементом проектирования систем, способных анализировать данные, выявлять закономерности и принимать решения на основе полученных знаний.

Системы искусственного интеллекта, основанные на машинном обучении, преодолевают ограничения традиционных программных подходов, позволяя компьютерам обучаться на основе опыта и самостоятельно совершенствовать свои алгоритмы работы. Открытые программы и библиотеки, такие как TensorFlow, PyTorch, scikit-learn и другие, предоставляют разработчикам мощные инструменты для создания и настройки моделей машинного обучения.

Проектирование систем искусственного интеллекта с использованием открытых программ позволяет сократить время и затраты на разработку, так как разработчики могут использовать уже готовые решения и алгоритмы, адаптируя их под конкретные задачи. Этот подход также способствует созданию более открытых и прозрачных систем, поскольку исходный код открытых программ доступен для общественного изучения и проверки.

Однако успешное применение открытых программ для решения задач машинного обучения требует не только технических знаний, но и глубокого понимания принципов работы алгоритмов машинного обучения и особенностей конкретной предметной области. Кроме того, важным аспектом является непрерывное обновление и совершенствование моделей на основе новых данных и отзывов от пользователей.

Таким образом, проектирование систем искусственного интеллекта с использованием открытых программ для решения задач машинного обучения представляет собой увлекательное и перспективное направление развития информационных технологий, способное принести значительные выгоды как бизнесу, так и обществу в целом.

Системы обучения "с учителем"

Системы обучения "с учителем" являются ключевым компонентом в машинном обучении и представляют собой метод, при котором модель обучается на основе предоставленных данных, снабженных метками или ответами. Этот подход позволяет модели "учиться" на основе известных примеров и использовать их для деления новых данных или прогнозирования результатов.

Процесс обучения "с учителем" начинается с построения модели, которая затем обучается на обучающем наборе данных. Этот набор данных содержит пары входных признаков и соответствующих им выходных значений или меток. В процессе обучения модель адаптируется к данным, строящим связь между входами и выходами, чтобы предсказать правильные ответы на новых данных.

Одним из основных методов в системах обучения "с учителем" является классификация, при которой модель предсказывает категориальные метки или классы для заданных входных данных. Например, модель может классифицировать электронные письма как "спам" или "не спам", изображения как "кошка" или "собака" и т. д.

Преимущества систем обучения "с учителем" включают их способность к точному прогнозированию на основе известных данных, возможность интерпретации результатов и широкое применение в различных областях, таких как финансы, медицина, технологии и другие. Однако эффективность модели "с учителем" часто зависит от качества и разнообразия обучающих данных, а также от правильного выбора алгоритмов и параметров модели.

|  |
| --- |
|  |

Нами были выбраны для задачи определения машины на фотографии по нескольким причинам:

• **Распознавание образов:** Нейронные сети превосходно распознают сложные образы, в том числе изображения машин, несмотря на различные углы обзора, освещение и фоны.

• **Автоматическое извлечение признаков:** Нейронные сети могут автоматически извлекать из изображений важные признаки, связанные с машинами, без необходимости вручную определять эти признаки.

• **Точность:** Нейронные сети, обученные на больших наборах данных изображений машин, могут достигать высокой точности при определении машин на новых изображениях.

• **Обобщение:** Нейронные сети могут обобщать свои знания и определять машины в различных условиях, даже если они не встречались в тренировочном наборе данных.

• **Удобство:** Существуют предустановленные нейронные сети, такие как ResNet или VGG19, которые были разработаны и обучены для задач распознавания изображений, что упрощает реализацию задачи по определению машин.

Кроме того, использование нейронных сетей позволяет реализовать более глубоки и сложные задачи, как:

• **Локализация:** Определение не только наличия машины на изображении, но и ее точного местоположения.

• **Классификация:** Определение типа машины (например, автомобиль, грузовик, мотоцикл).

• **Детализация:** Извлечение дополнительных сведений о машине, таких как марка, модель или цвет. Нейронные сети обеспечивают гибкий и эффективный подход к задаче определения машин на фотографиях, что делает их идеальным выбором для этой задачи.

**Решение проблемы классификации**

Проблема классификации в машинном обучении заключается в назначении входных данных к одной из нескольких предопределенных категорий, известных как классы. Цель состоит в том, чтобы научить модель, которая может предсказывать класс новых, невидимых данных с высокой точностью. Проблема классификации возникает во многих реальных приложениях, таких как:

• Распознавание изображений (например, классификация животных, транспортных средств, объектов)

• Обработка естественного языка (например, классификация текстов по темам, настроениям, языкам)

• Финансовый анализ (например, классификация кредитных заявок на высокорисковые и низкорисковые)

• Медицинская диагностика (например, классификация медицинских изображений как нормальных или патологических)

При решении проблемы классификации можно столкутся с несолькими проблемами: **переобучение, нехватка данных, дисбаланс классов, высокая размерность данных**.

**Переобучение:** Модель может слишком хорошо подойти к тренировочным данным и плохо обобщить новые данные.

**Нехватка данных:** Наличие недостаточного количества данных для обучения эффективной модели.

**Дисбаланс классов:** Распределение классов в наборе данных может быть неравномерным, что приводит к смещению модели в сторону классов с большим количеством данных.

**Высокая размерность данных:** Данные могут иметь множество признаков, что может усложнить обучение модели и привести к проблемам с производительностью.

Решая эти проблемы, можно разработать надежные модели классификации, которые могут эффективно назначать входные данные к соответствующим классам.

Проблема классификации машин на фотографиях была решена с помощью следующих шагов:

**1. Сбор данных:**

• Собрали большой набор данных изображений машин с различными типами, углами обзора, освещением и фонами.

• Аннотировали изображения вручную, указав тип машины на каждом изображении.

**2. Предварительная обработка данных:**

• Изменили размер изображений для обеспечения единообразия.

• Нормализовали пиксельные значения для улучшения сходимости нейронной сети.

**3. Выбор и настройка нейронной сети:**

• Выбрали предварительно обученную нейронную сеть, специализирующуюся на распознавании изображений, например ResNet или VGGNet.

• Настроили нейронную сеть для задачи классификации машин, добавив выходной слой с количеством узлов, соответствующим количеству классов машин.

**4. Обучение нейронной сети:**

• Разделили набор данных на тренировочный и тестовый наборы.

• Обучили нейронную сеть.

**5. Оценка модели:**

• Оценили производительность обученной нейронной сети на тестовом наборе.

• Вычислили метрики точности, такие как точность и потеря данных, чтобы определить эффективность классификации.

**Также мы использовали некоторые другие приемы:**

• Раннее прекращение обучения, чтобы предотвратить переобучение.

После завершения этих шагов у нас есть обученная нейронная сеть, которая может классифицировать машины на фотографиях с высокой степенью точности.

**Выбор**

Для реализации поставленной задачи нам необходимо выбрать несколько архитектур для нейросети.

Изображение выглядит как текст, число, меню, документ

Автоматически созданное описание

Ориентируясь на составленную таблицу с данными об архитектурах мы выбрали 7 наиболее подходящих для классификации наличия машины на фотографии или ее отсутствии.

**Лучшие 7 моделей для определения машины на картинке:**

1. **Xception** - мощная модель с детальной блочной факторизацией, обеспечивающая высокую точность классификации.

Изображение выглядит как диаграмма, План, Технический чертеж, линия

Автоматически созданное описание

1. **VGG19** - глубокая модель с последовательностью сверточных и пулинговых слоев, известная своей универсальностью и хорошей производительностью в различных задачах компьютерного зрения.

Изображение выглядит как снимок экрана, диаграмма, текст, График

Автоматически созданное описание

1. **ResNet50** - модель на основе глубокой остаточной сети с прямым соединением между входами и выходами слоев, что улучшает градиентный поток и точность.

Изображение выглядит как диаграмма, линия, снимок экрана, зарисовка

Автоматически созданное описание

1. **InceptionV3** - модель со сложной архитектурой, состоящей из нескольких параллельных веток сверточных операций, обеспечивающая высокую точность за счет большего количества параметров и вычислительной емкости.

Изображение выглядит как диаграмма, снимок экрана, дизайн

Автоматически созданное описание

1. **EfficientNetB4** - модель, оптимизированная для эффективности и точности, обеспечивающая баланс между производительностью и вычислительными затратами.

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

6. **ResNet152** - более глубокая версия ResNet50 с еще более мощными возможностями для извлечения признаков и более высокой точностью.

Изображение выглядит как снимок экрана, текст, диаграмма, Шрифт

Автоматически созданное описание

7. **ConvNeXtTiny** - модель с трансформерами свертки, которая представляет собой гибрид CNN и трансформеров, обеспечивая улучшенную точность на более легких и быстрых моделях.

Изображение выглядит как текст, диаграмма, снимок экрана, План

Автоматически созданное описание

Все из выбранных 7 нейросетей подходят для задачи распознавания и классификации изображений

**Выбор 3**

Для отбора наилучших 3 нейросетей из выбранных 7 для детального рассмотрения мы составили таблицу для этих архитектур.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

При детальном рассмотрении и изучении всех архитектур мы решили остановить свой выбор на 3 из них:

1. Xception:

- Предлагает глубокую и мощную архитектуру с глубиной 140 слоев.

- Достигает высокой точности при классификации изображений и обнаружении объектов.

- Уменьшает объемы вычислений по сравнению с Inception и ResNet, делая его более подходящим для мобильных устройств и встраиваемых систем.

2. VGG: - Один из самых простых и широко используемых архитектур CNN.

- Имеет глубокую архитектуру с несколькими стеками сверточных слоев.

- Обеспечивает хорошую точность при обнаружении объектов и сегментации изображений.

- Является надежной основой для многих современных архитектур CNN.

3. Inception:

- Представляет собой сложную вложенную структуру, использующую многомасштабные свертки для извлечения особенностей из изображений.

- Достигает выдающейся точности при классификации изображений и обнаружении объектов. - Более вычислительно дорогая, чем VGG, но обеспечивает более высокие показатели.

Эти архитектуры прекрасно они предлагают оптимальное сочетание факторов. **Точности, мощности, глубины и надежности. Также используя эти модели, мы изучили различные архитектурные подходы что поможет нам взглянуть на данную задачу с разных сторон и выбрать разные методы ее решения для достижения наилучшего результата.** Xception и Inception имеют умеренное количество параметров и в то же время высокую точность, что делает их подходящими для задач с ограниченными вычислительными ресурсами. VGG имеет больше параметров, но также обеспечивает высокую точность, хотя и требует более высоких вычислительных затрат.

После выбора подходящих нам архитектур, нам нужно было подготовить набор данных для машинного обучения. С самого начала мы определили два класса для решения задачи распознавания: Изображения с автомобилями, изображения без автомобилей Для формирования обучающих и тестирующих выборок. Мы фотографировали машины в дневной время, для обеспечения достаточной видимости машины на фотографии. Фотографии с отсутствие машины на фотографии были подобраны так, чтобы всеми факторами, как освещением, так и ее содержанием, отличались от фотографий с машинами. Далее все данные были разделены на обучающую и тестирующую выборки в соотношении 80:20, что является стандартным соотношением для задач машинного обучения. Для обучающей выборки было отобрано 300 изображений, из которых:

•150 изображений содержат автомобили

• 150 изображений не содержат автомобилей

Для контрольной выборки было отобрано 50 изображений, из которых:

• 25 изображений содержат автомобили

• 25 изображений не содержат автомобилей

Изображения для обучения и контроля были отобраны вручную для обеспечения разнообразия объектов, ракурсов и условий освещения.

Изображение выглядит как колесо, Наземный транспорт, транспортное средство, на открытом воздухе

Автоматически созданное описание

пример фотографий с машиной

Изображение выглядит как Человеческое лицо, человек, очки, зрелище

Автоматически созданное описание

Пример фотографии без машины

Следующая задача, которая у нас стояла, это обеспечение подходящего расширение для работы нейросети с нашим набором данных. Все выбранные нами нейросети работают с расширением 299x299. Для быстрого уменьшения расширения была написана программа(Приложение 1). После чего фотографии изменились для обеспечения работы нейросети с ними.

Изображение выглядит как Человеческое лицо, человек, очки, зрелище

Автоматически созданное описание

Фотография 299x299

Изображение выглядит как колесо, транспортное средство, Наземный транспорт, на открытом воздухе

Автоматически созданное описание

Фотография 299x299

**Описание разработки модели распознавания образов**

**Программа**

Далее была написана модель распознавания образов на основе обучающей выборки. Модель была написана, используя открытые библиотеки Pyton: Keras и Tensorflow.

Для определения оптимального количества эпох обучения использовался график зависимости достигнутой точности обучения от числа эпох. Обучение проводилось с использованием различных значений количества эпох, и график был построен на основании полученных результатов. Точка, в которой прекращается прирост точности обучения, была использована для выбора оптимального количества эпох.

**Loss (потери)** в нейросети — это мера того, насколько плохо модель предсказывает выходные данные для данного набора входных данных. Чем меньше значение потери, тем лучше модель предсказывает выходные данные. Представьте, что вы учите нейросеть распознавать кошек на изображениях. Вы показываете сети множество изображений кошек и не кошек, а затем Потери — это способ количественно оценить, насколько неточны предсказания сети. Чем больше ошибок делает сеть, тем выше потери. Цель обучения нейронной сети - минимизировать потери. Это делается путем настройки весов и смещений сети до тех пор, пока она не начнет делать точные предсказания. По мере обучения потери будут уменьшаться, что указывает на то, что сеть становится лучше в распознавании кошек.

Количество нейронов в каждом слое, а также глубина сети были определены экспериментально. Модели с различным количеством нейронов, а также различной глубиной были обучены на наборе данных, и модель с наилучшими показателями точности была выбрана для дальнейшей работы. Модель состояла из следующих слоев:

• Сверточные слои: Извлечение признаков из входных данных.

• Слои объединения: Уменьшение размерности данных.

• Полносвязные слои: Классификация признаков.

**Обоснование выбора параметров:**

Оптимальные параметры были выбраны на основании высокой точности обучения на наборе данных, а также модель должна быть эффективной для обучения и использования. Выбранные параметры обеспечивают наилучший баланс между этими критериями, что приводит к эффективной и точной модели распознавания образов.

**Анализ качества обучения**

Смысловая нагрузка критериев оценки качества обучения:  
Точность измеряет долю правильно классифицированных образцов в наборе данных. Высокая точность означает, что модель хорошо различает разные классы объектов.  
Потери измеряют разницу между прогнозами модели и фактическими значениями. Низкие потери означают, что модель хорошо подходит данным.

|  |  |  |
| --- | --- | --- |
|  | Точность | Потери |
| VGG19 | 0,9442 | 0,1600 |
| Xception | 1,0000 | 0,0006 |
| InceptionV3 | 1,0000 | 0,0010 |

На таблице №……… видно что наивысшую точность на обучающей выборке показали нейросети на основе модели Xception и InceptionV3. Нейросеть на модели VGG19 также показала достаточно высокую точность. Потери нейросетей Xception и InceptionV3 также оказались достаточно низкими, по сравнению с VGG19.

**Анализе качества распознавания**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Xception | InceptionV3 | Vgg19 |
| Кол-во объектов | 100 | 100 | 100 |
| Кол-во объектов с машинами | 50 | 50 | 50 |
| Кол-во объектов без машин | 50 | 50 | 50 |
| Кол- во верно распознанных | 97 | 59 | 96 |
| кол во ошибок | 3 | 41 | 4 |
| Кол-во ошибок 1 рода | 3 | 28 | 2 |
| Кол-во ошибок 2 рода | 0 | 13 | 2 |
| % Точности | 97% | 59% | 96% |
| % Ошибок 1 рода | 6,00% | 56,00% | 4,00% |
| % Ошибок 2 рода | 0,00% | 26,00% | 4,00% |

Для обучающей выборки были отобраны 100 объектов по 50 объектов с машинами и 50 без машин. В результате тестирования можно заметить, что наименьшее кол-во ошибок и наибольшую точность показала нейросеть с моделью Xception. Нейросеть VGG19, которая на этапе обучения не отличалась высокой точность и минимальными потерями по-сравнению со своими конкурентами, показала достаточно хорошие показатели – 96% точности. Нейросеть, обучающаяся дольше остальных на модели InceptionV3, показала худший результат из всех. Наибольшее кол-во ошибок данной нейросети приходится на ошибки 1 рода. Эта нейросеть часто определяла данные с классом ‘’Машины’’ в класс ‘’Не машины’’. Немного меньше нейросеть ошибается при ошибках 2 рода.

На основе обучения и изучения моделей и нейросетей и их архитектур

Приложение 1

import cv2  
import os  
import os.path  
*# Папка с вашими исходными изображениями*source\_folder = "D://Users//USER//naxui//test//no\_car//"  
*# Папка, в которой будут сохранены измененные изображения*target\_folder = "D://Users//USER//naxui//lol"  
*# Желаемый размер изображений*desired\_size = (299, 299)  
  
*# Создание целевой папки, если она не существует*if not os.path.exists(target\_folder):  
 os.makedirs(target\_folder)  
  
*# Перебор всех изображений в папке*for filename in os.listdir(source\_folder):  
 *# Чтение изображения* mapp = os.path.join(source\_folder, filename)  
 image = cv2.imread(mapp)  
 *# Проверка, было ли успешно загружено изображение* if image is not None:  
 *# Изменение размера изображения* resized\_image = cv2.resize(image, desired\_size)  
 *# Сохранение измененного изображения в целевую папку* cv2.imwrite(os.path.join(target\_folder, filename), resized\_image)  
 else:  
 print(f"Ошибка загрузки изображения: {filename}")  
  
print("Процесс завершен.")