1. Тема моей работы: «Проектирование и разработка приложения для интеллектуальной обработки изображений»

2. Целью работы было создать нейронную сеть и веб-приложение для нее, позволяющее в автоматическом режиме проводить повышение качества изображений, такие как увеличение разрешения, колоризация, сегментация (например, для вырезания фона).

3. Наборы данных были взяты с сайта kaggle, на нем в открытом доступе имеется огромное количество разнообразных наборов данных.

Были выбраны два набора, содержащие пары изображений четких и не очень, как в первом случае, и цветные и ч/б как в случае второго набора.

Они уже были подготовлены к обучению, то есть имели одинаковый размер и цветовой формат, и, как можно заметить они все имеют крайне маленькое разрешение: 255 в случае первого набора и всего 160 в случае второго.

Для обучения нейронных сетей часто используются изображения маленького размера по нескольким причинам:

* снизить требования к вычислительной мощности.
* Обучение нейронных сетей на маленьких изображениях может способствовать упрощению модели, снижению сложности задачи и весу модели. Маленькие изображения содержат меньше деталей и особенностей, поэтому модель может сосредоточиться на обнаружении более общих паттернов и признаков.

Модель, обученная на маленьких изображениях, может иметь затруднения с обработкой больших или более детализированных изображений. В данном случае это будет значить, что изображение, полученное от пользователя нужно будет сначала привести к размеру 255 или 160 пикселей, так как модели будут ожидать на вход именно такой размер.

4. Вся нейронная сеть, представленная в данной работе, реализована с помощью трех основных инструментов: язык Python, библиотеки TensorFlow и Keras. Язык пайтон идиционна используется для такого рода задач.

Если кратко сказать про TensorFlow, то он организует всю логику работы нейросети. Керас работает поверх ТФ и упрощает работу с ним.

React сейчас является популярнейшей библиотекой для создания одностраничных приложений, и он весьма прост в освоении. Вместе с ним часто используется express для создания rest api для приложения, он также популярен и удобен, поэтому выбор пал на эти две технологии.

5. При создании нейронных сетей для работы с изображениями, как правило, применяется свертка. Предполагается, что пиксели, находящиеся близко друг к другу, теснее “взаимодействуют” при формировании интересующего нас признака.

На каждом шаге, фильтр умножается на соответствующие пиксели изображения и суммирует результаты. Эта операция позволяет выделить локальные признаки, такие как границы, текстуры и углы.

Макспулинг - это операция, которая выполняется после свертки. Она помогает уменьшить размерность изображения и сделать представление более инвариантным к небольшим смещениям объектов на изображении. Для каждой области (обычно квадрата) изображения, выбирается максимальное значение и передается на следующий слой. Это позволяет уменьшить размер изображения, сохраняя важные признаки.

Свертка и максимальное пулинг обычно применяются последовательно несколько раз в сверточных слоях нейронной сети, чтобы постепенно извлечь все более высокоуровневые и абстрактные признаки из изображения.

6. Важную роль играет хорошая настройка гиперпараметров, основные из них представлены на слайде. Они управляют корректировкой весов в процессе обучения, могут вызывать раннюю остановку обучения для предотвращения переобучения, это когда модель хорошо классифицирует только примеры из обучающей выборки, управляют размером партии, т.е. числом экземпляров на каждой итерации и самим числом итераций или эпох.

Градиент - это вектор, который показывает направление наибыстрейшего роста функции в данной точке. Градиент используется для определения направления, в котором следует обновлять параметры модели для минимизации функции потерь.

Здесь скорость обучения (learning rate) - это гиперпараметр, который определяет величину шага, делаемого в направлении антиградиента. Слишком большое значение скорости обучения может привести к расхождению процесса оптимизации, а слишком маленькое значение может замедлить обучение.

7. Для решения задач, поставленных в данной работе, была выбрана архитектура U-Net. U-Net – это сверточная нейронная сеть, которая была разработана для сегментации биомедицинских изображений. Архитектура сети представляет собой полносвязную свёрточную сеть, модифицированную так, чтобы она могла работать с меньшим количеством примеров и делала более точную сегментацию.

Архитектура U-Net состоит из двух основных частей: сжимающего пути (encoder) и расширяющего пути (decoder).

Сжимающий путь состоит из сверточных слоев, а также слоев объединения (pooling layers), которые позволяют уменьшить размерность входных изображений и выделить важные признаки. Затем эти признаки передаются в расширяющий путь.

Расширяющий путь состоит из слоев деконволюции (deconvolution layers) и слоев объединения (upsampling layers), которые позволяют увеличить размерность входных данных и восстановить карту сегментации. В конце расширяющего пути добавляется последний слой, который используется для получения итоговой карты сегментации.

8. Процесс обучения. Тут можно увидеть поэтапно процесс обучения двух нейронных сетей.

1. Веса нейронной сети инициализируются случайными значениями.
2. Сравнивается предсказанное цветное изображение с фактическим цветным изображением. Разница между ними измеряется с помощью функции потерь.
3. Градиенты функции потерь вычисляются по отношению к весам нейронной сети с помощью обратного распространения ошибки. Градиенты показывают, как изменение весов влияет на значение функции потерь.
4. С использованием градиентов, вычисленных на предыдущем шаге, обновляются веса нейронной сети с целью минимизации функции потерь. Этот шаг повторяется множество раз, чтобы нейронная сеть могла находиться в состоянии, в котором её предсказания будут наиболее близкими к фактическим цветным изображениям.

9. Но как понять сколько эпох нужно для наилучшей точности? Можно взглянуть на график потерь для наборов для обучения и валидации. Потери (loss) представляют собой функцию, которая измеряет расхождение между предсказанными значениями модели и истинными значениями из набора для обучения.

График для набора для обучения всегда стремится вниз, а для валидационного набора, как правило, вверх (чем выше – тем хуже). Нужно обучить сеть так, что-бы она одинаково хорошо работала как на тренировочном наборе, так и на реальных данных. На этом графике это 5-6 эпоха.

Для моделей из этой работы длительность обучения на средней видеокарте составила 10-15 минут, вес итоговых моделей порядка 100 мб.

10-11. Результаты.

12. Приложение. В приложении можно выбрать тип обработки, загрузить файл с изображением и сохранить результаты.

13. Заключение. В ходе выполнения работы были осуществлены исследование, проектирование и разработка приложения обработки изображений. Основная цель заключалась в создании инструмента, способного автоматизировать процесс обработки и анализа изображений, обеспечивая точность, эффективность и удобство использования.

Использование программ, таких как представленная в данной работе, не требует от пользователя никаких прикладных знаний в области обработки изображений и существенно экономит время при использовании, относительно работы с графическими редакторами.

Направлением для дальнейшей работы может быть увеличение качества обработки путем изменения модели, гиперпараметров или увеличения набора данных, расширение функционала приложения, например, возможность создать аккаунт и сохранять изображения на сервере.