

Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Компьютерных сетей и систем

Кафедра Информатики

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

по курсу «Машинное обучение»

Реализация приложения по распознаванию номеров домов

Студент:
гр. 758641
Ярош Г.И.

Проверил:
Заливако С. С.

Минск, 2019

СОДЕРЖАНИЕ

РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ ПО РАСПОЗНАВАНИЮ НОМЕРОВ ДОМОВ 3

1. Цель	3
2. Набор данных SVHN	3
3. Классификация данных второго типа	4
4. Классификация данных первого типа	8
5. Приложение для OS Android	11
6. Вывод	13
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	14

РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ ПО РАСПОЗНАВАНИЮ НОМЕРОВ ДОМОВ

1. Цель

Изучить структуру архива SVHN, содержащую изображения номеров домов. Построить глубокую нейронную сеть для распознавания цифр в номерах домов. Реализовать приложение для OS Android, которое будет способно распознавать цифры в номерах домов используя глубокую нейронную сеть.

2. Набор данных SVHN

Архив SVHN содержит данные двух типов. Данные первого типа содержат фотографии номеров домов в большом разрешении (рис. 1). Номера домов на центрированы и не выделены. Данные второго типа представляют собой набор изображений номеров домов размером 32x32, полученный из набора данных первого типа путем обрезки номера домов по изначально размеченным рамкам (рис. 2).



Рис. 1. Пример изображений первого типа



Рис. 2. Пример изображений второго типа

Для обоих типов данных архив содержит три выборки: тренировочную, тестовую и дополнительную.

3. Классификация данных второго типа

В данной работе было проведено исследование возможности классификации реальных изображений, если классификатор обучить на синтетических данных. В качестве таких синтетических данных был выбран архив MNIST (рис. 3). Для соответствия размеров изображений из архива MNIST и SVHN, изображения MNIST были приведены к разрешению 32x32, а изображения в SVHN были приведены к черно-белому виду.

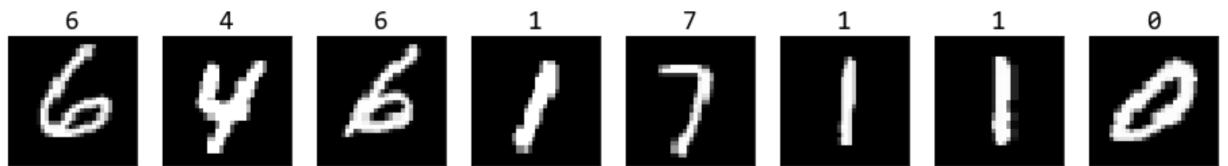


Рис. 3. Пример изображений из архива MNIST.

В качестве модели для классификации мною была построена глубокая сверточная сеть, состоящая из трех сверточных слоев и слоев дискретизации после каждого из них. За сверточными слоями следует полно связанный слой. Функция всех скрытых слоев – кусочно-линейная. Выходной слой также полно связанный с функцией активации softmax. Архитектура сети отображена на рисунке 4.

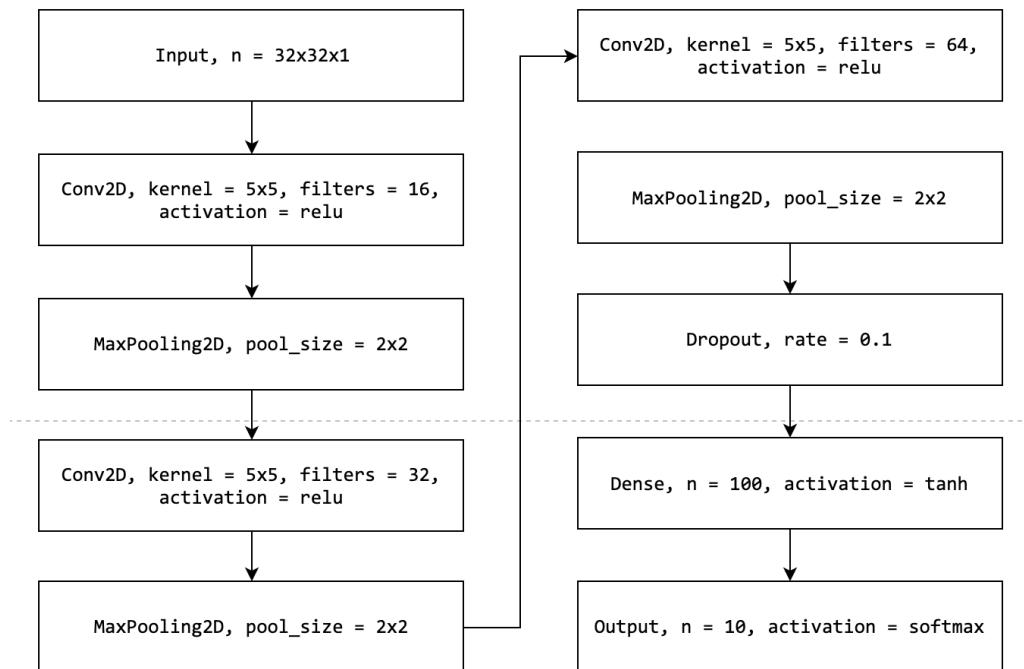


Рис. 4. Архитектура глубокой сверточной сети.

Из тренировочной выборки была выделена валидационная выборка в 10% исходной. Обучение нейронной сети производилось с помощью алгоритма оптимизации Adam. Размер батча был выбран в 100. Количество эпох без улучшения, при которых сеть заканчивала обучаться, - 10. В качестве функции потерь использовалась категориальная кроссэнтропия. В качестве метрики точности – категориальная точность.

Зависимости значения функции потерь и точности для обучающей и валидационной выборок представлены на рисунках 5 и 6.

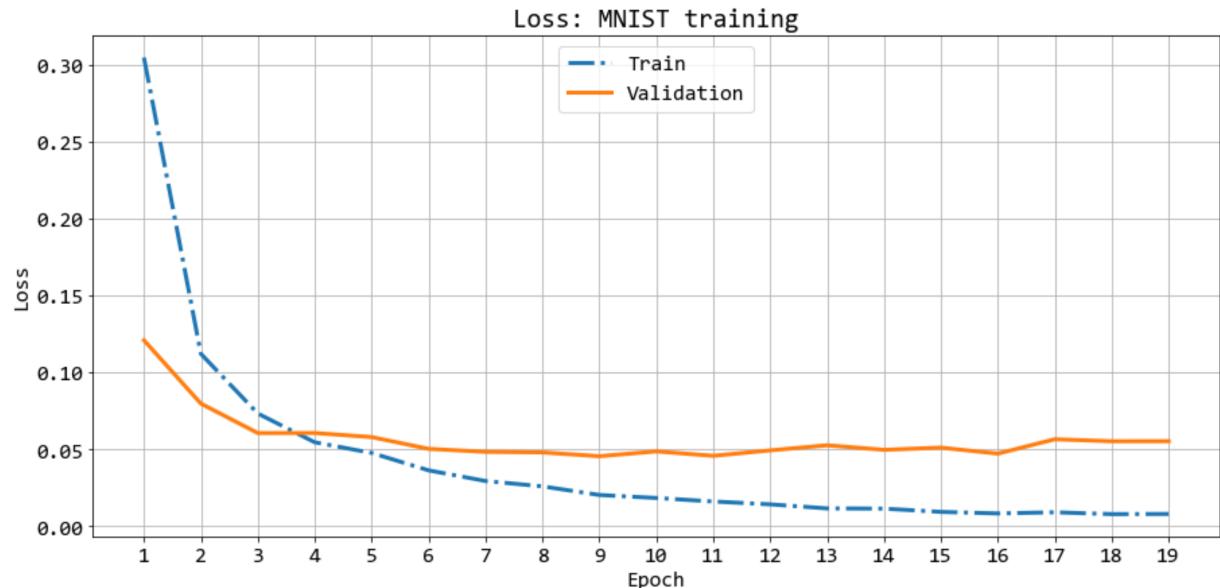


Рис. 5. Уменьшение значения функции потерь при обучении.

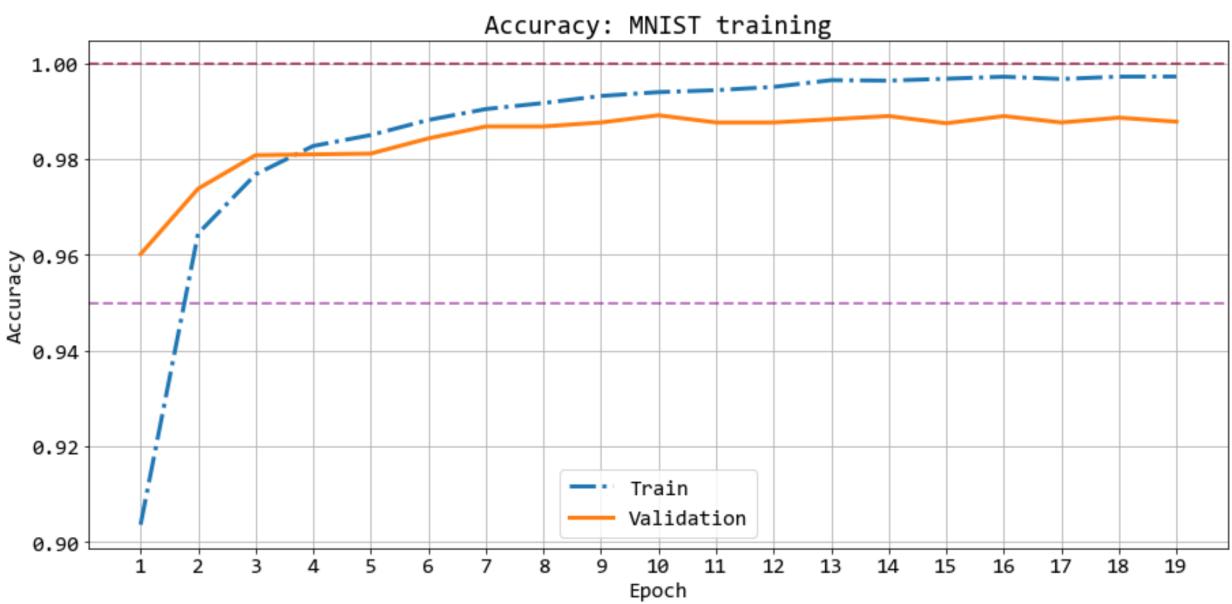


Рис. 6. Увеличение точности классификации при обучении.

В результате, после 19 эпох, обучение нейронной сети остановилось. Точность классификации на тестовой выборке SVHN составила всего лишь 0.23, что является полностью неудовлетворительным результатом.

Далее было проведено обучение модели на данных из архива SVHN. Изменения значения функции потерь и точности классификации приведены на рисунках 7 и 8.

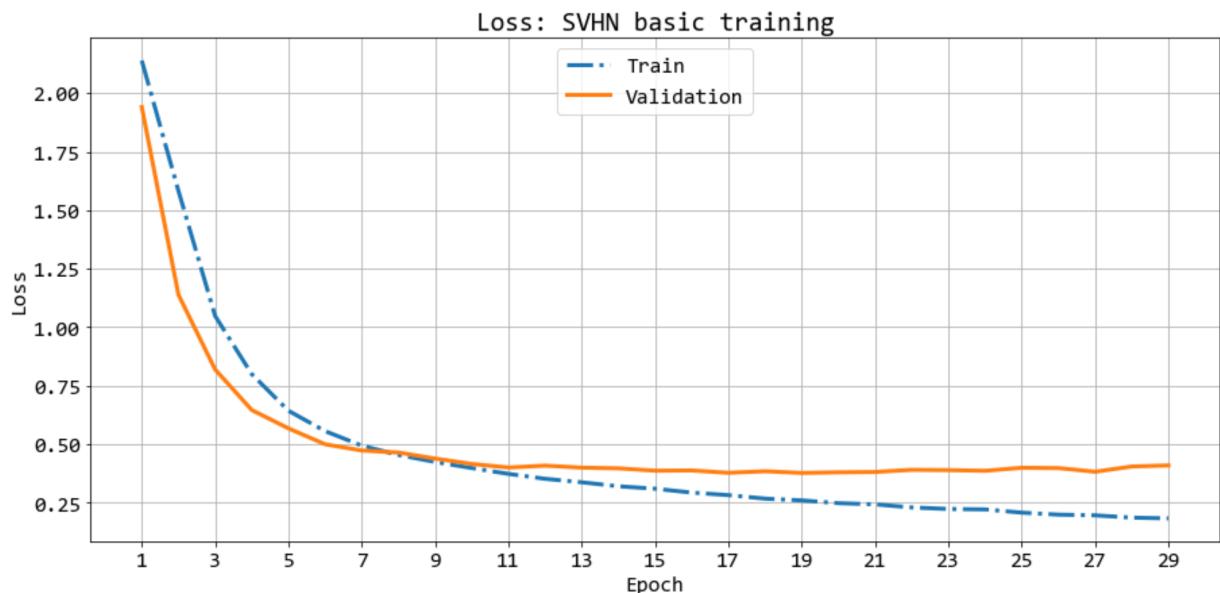


Рис. 7. Изменения функции потерь в процессе обучения.

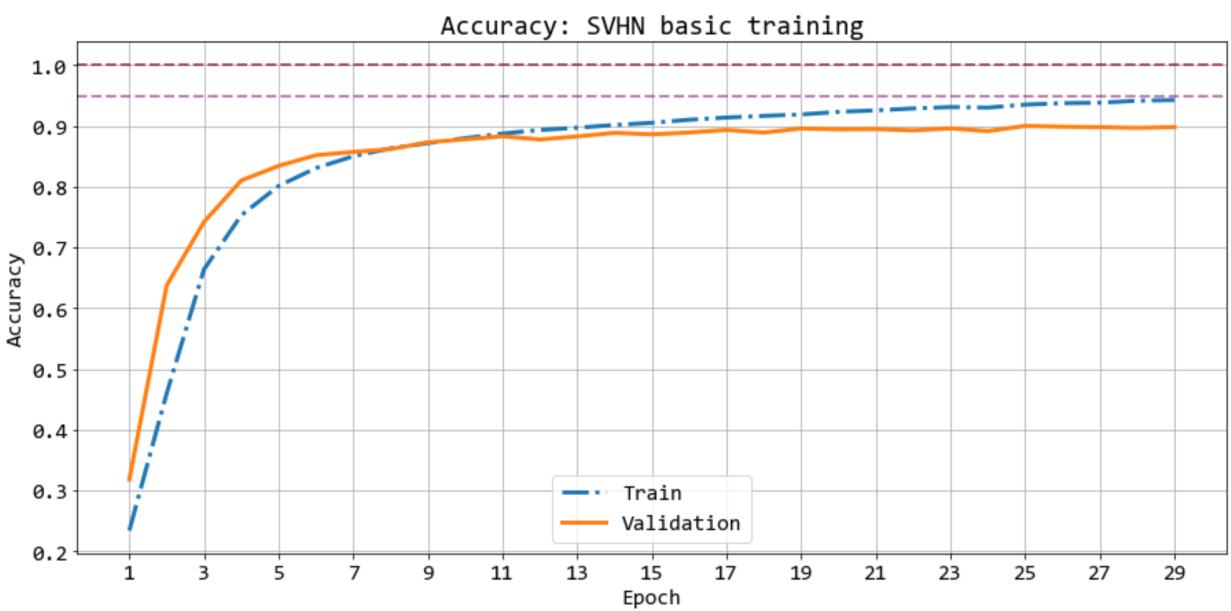


Рис. 8. Изменения точности классификации в процессе обучения.

Обучение завершилось после 29 эпох. Точность классификации этой модели на тестовых данных SVHN составила 0.89, что уже неплохой показатель.

Для улучшения точности модели она была дообучена на дополнительных данных из архива. Процесс обучения отображен на рисунках 9, 10. Обучение завершилось после 20 эпохи. Точность классификации модели обученной на дополнительной выборке составила 0.936.

Сравнение точности всех моделей, которые были обучены на втором типе данных из архива notMNIST представлено на рисунке 11.

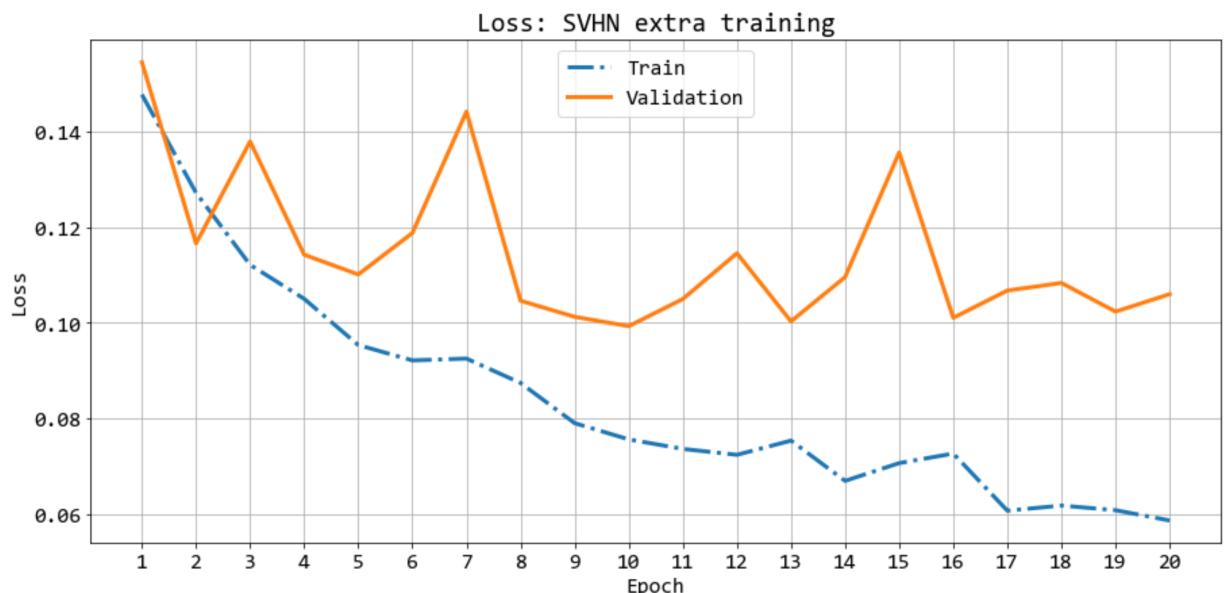


Рис. 9. Изменение функции потерь в процессе обучения.

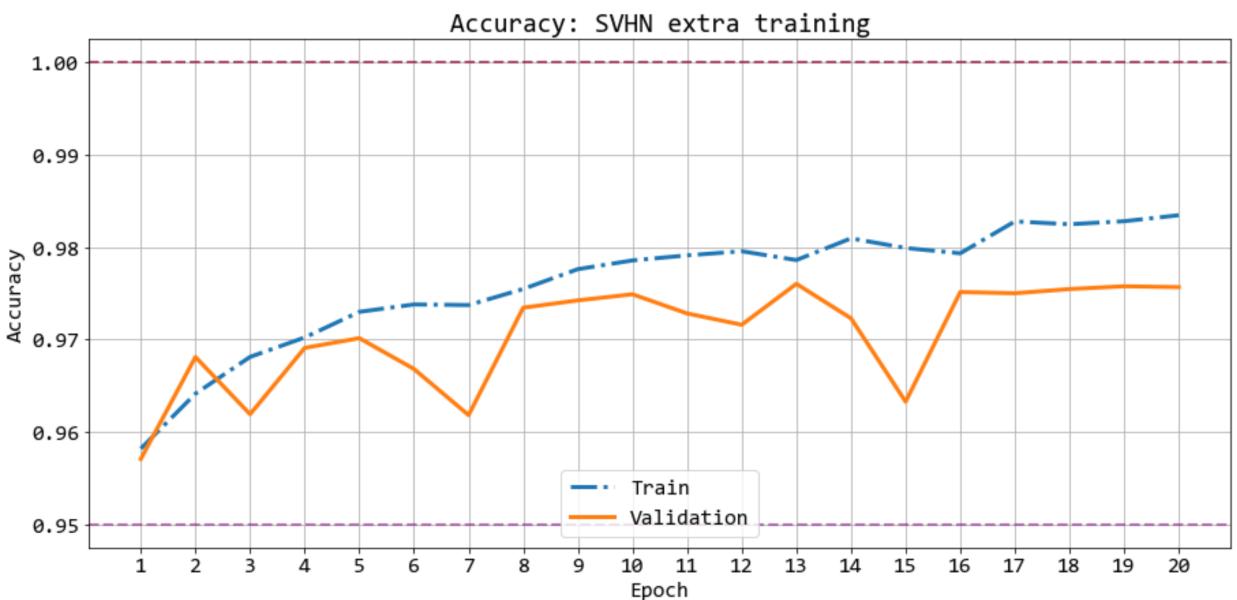


Рис. 10. Изменение точности в процессе обучения.

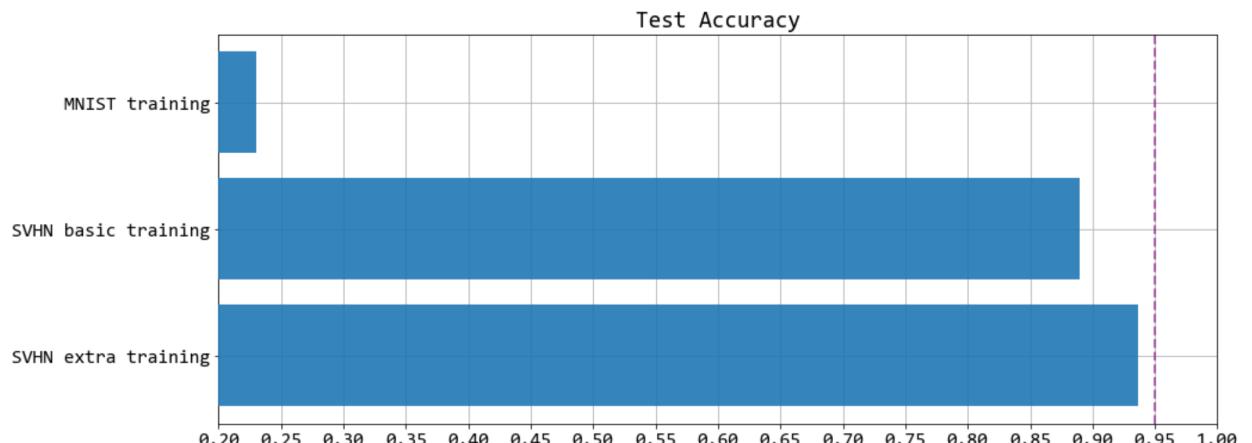


Рис. 11. Сравнение точности классификации моделей, полученных в данном пункте лабораторной работы.

4. Классификация данных первого типа.

Данные первого типа в архиве SVHN представляют собой полные изображения номеров домов в различном разрешении. Перед началом работы, все изображения были обрезаны по рамкам, которые выделяют номера домов на изображениях, таким образом, чтобы результирующее изображение содержало целый номер дома. Далее изображения были приведены к разрешению 96x96. Пример результирующих изображений приведен на рисунке 12.



Рис. 12. Пример изображений, обрезанных по рамкам цифр

Классы изображений были представлены в виде набора векторов длиной 11, каждый из которых соответствовал одной цифре изображения. Сами векторы были представлены в формате one-hot, где единица на i -том месте обозначала соответствовала классу цифры i , а 11-ый класс обозначал отсутствие цифры на данной позиции.

Далее был проведен анализ длин номеров домов в тестовой и тренировочной выборках (рис. 13, 14, таблица 1). Оказалось, что архив в основном содержит номера домов длиной в 1, 2 или 3 цифры. Длинны номеров домов в 4 представлена в небольшом количестве, длины 5 и 6 почти в архиве не представлены.

Длина	1	2	3	4	5	6
Количество в обучающей выборке	5137	18130	8691	1434	9	1
Количество в тестовой выборке	2483	8456	2081	146	2	0

Таблица 1. Распределения длин номеров домов в тренировочной и тестовой выборках.

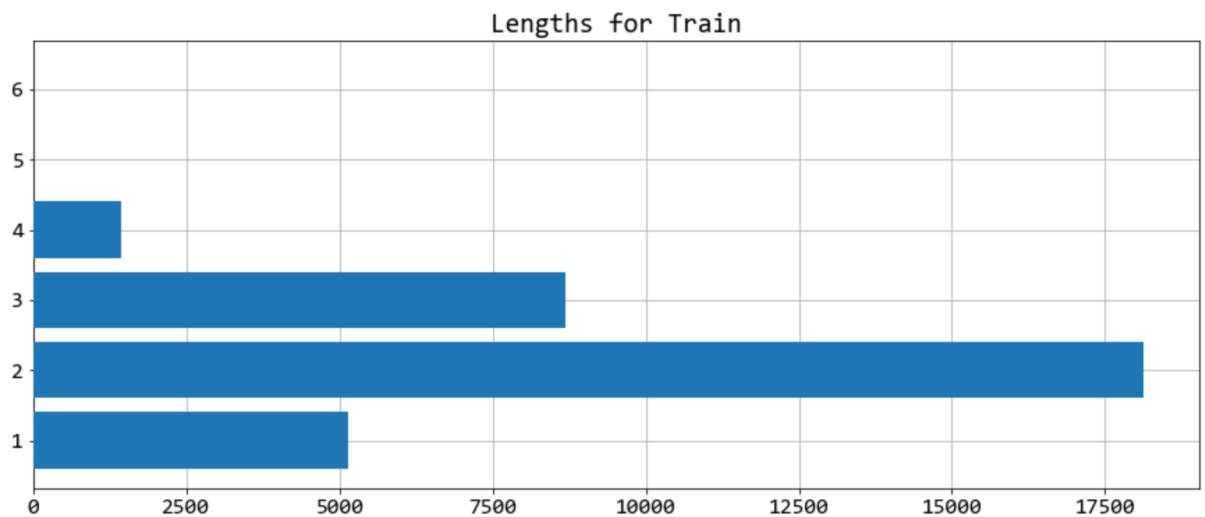


Рис. 13. Распределения длин в тренировочной выборке.

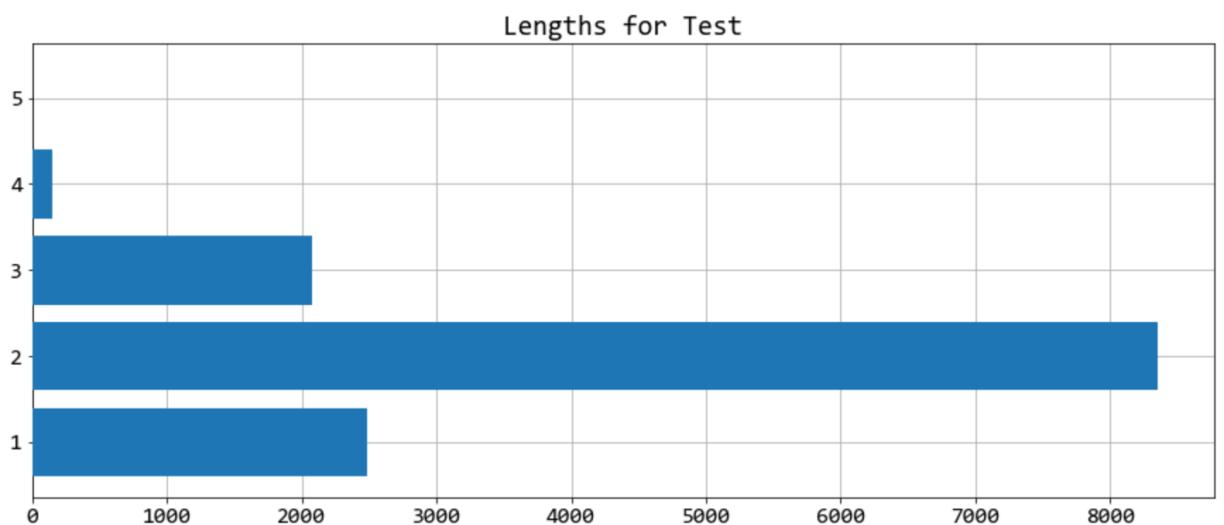


Рис. 14. Распределения длин в тестовой выборке.

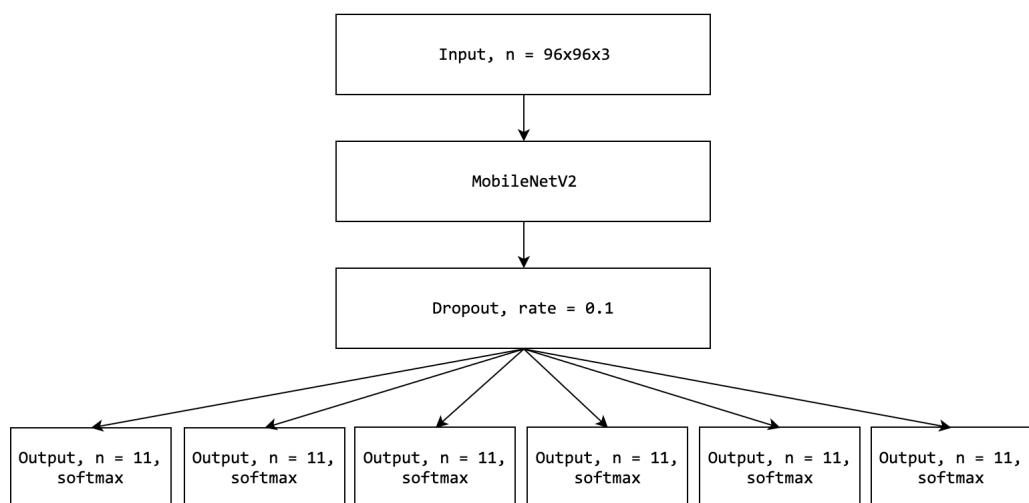


Рис. 15. Архитектура построенной сети.

В качестве модели для распознавания цифр использовалась готовая архитектура MobileNetV2. Она состоит из входного сверточного слоя с ядром 3x3,

17 специальных блоков (рис. 16) и выходного сверточного слоя с ядром 1x1. К данной сети я добавил 6 выходных полно связанных слоев длиной 11 и с функцией активации softmax (рис. 15).

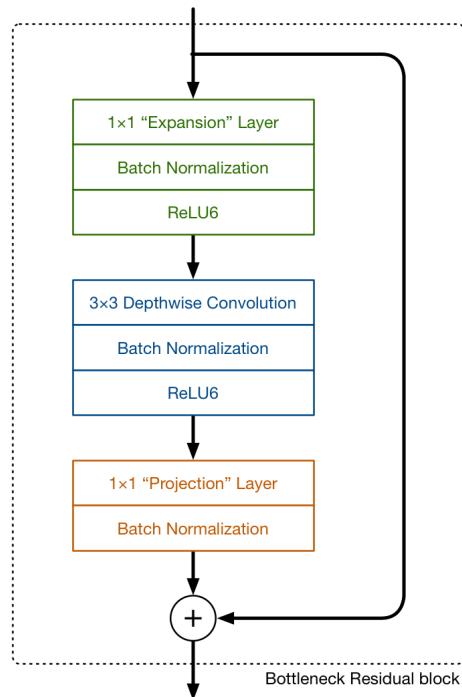


Рис. 16. Блок сети MobileNetV2.

Каждый выходной слой из 6 соответствовал цифре в номере дома. Выход со значением 1 для одиннадцатого класса означал отсутствие цифры на данной позиции.

Обучение производилось с помощью алгоритма оптимизации Adam. Функция потерь была задана как сумма категориальных кроссэнтропий от каждого выхода. Эффективность классификации при обучении измерялась с помощью категориальной точности для каждого выхода в отдельности. Обучение проводилось в два этапа. Сначала на тренировочной выборке. Затем модель была дообучена на дополнительной выборке из архива. Графики обучения можно увидеть на рисунках 17 и 18.

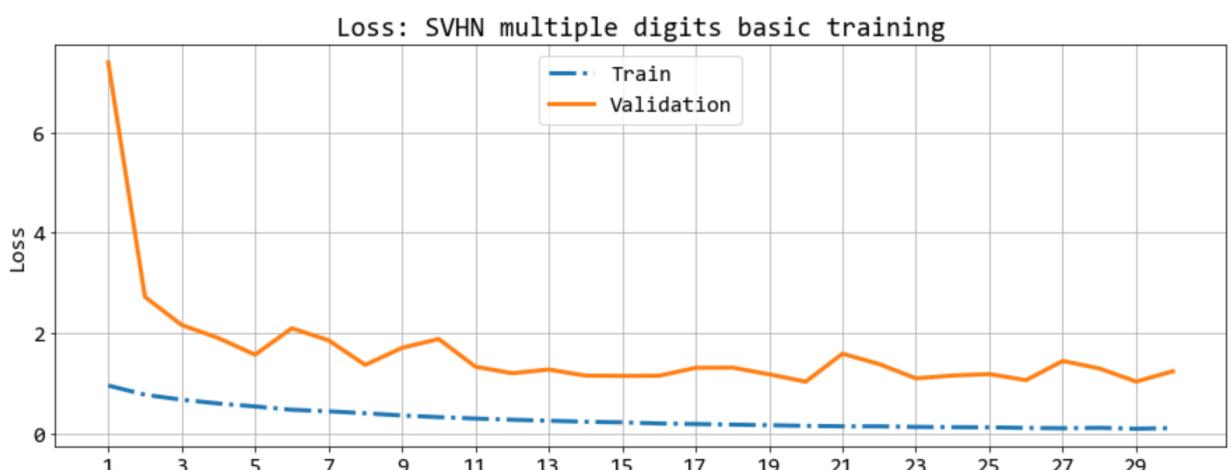


Рис. 17. Изменение значения функции потерь в процессе обучения.

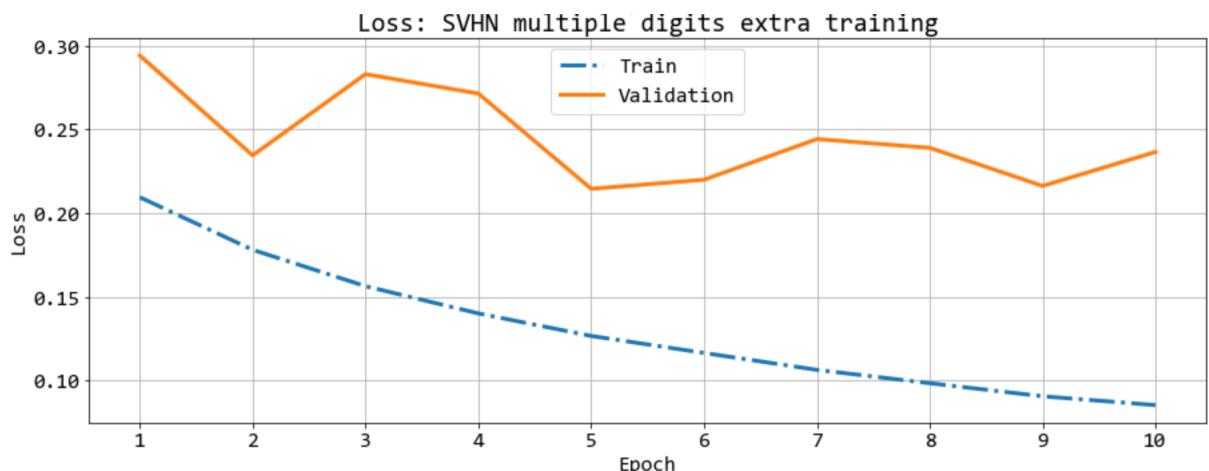


Рис. 18. Изменение функции потерь при обучении модели на дополнительной выборке.

Достигнутые точности модели после обучения на основной и дополнительной тренировочных выборках для всех цифр сразу приведены на рисунке 19.

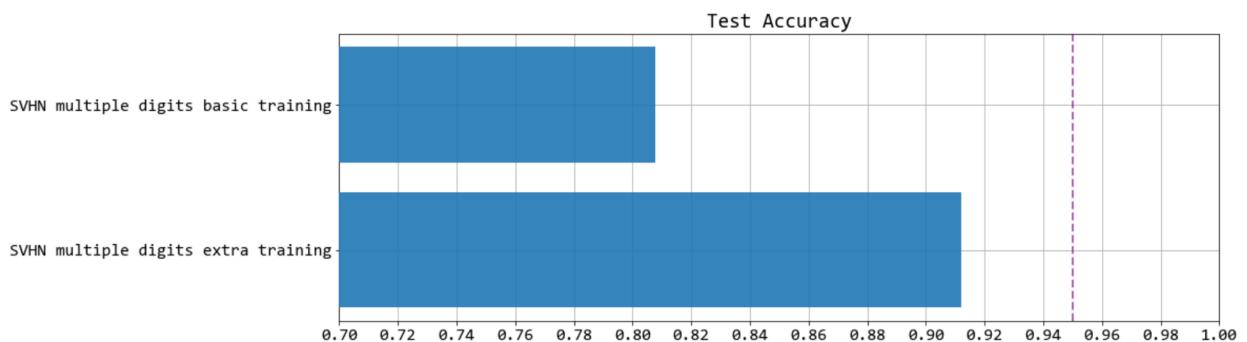


Рис. 19. Точности модели для тестовой выборки для всех цифр номеров домов.

Как мы видим, данная модель показала хорошую способность к классификации изображений с несколькими цифрами. Наибольшая точность составила 0.911.

5. Приложение для OS Android

В данной лабораторной работе также было разработано приложения под OS Android, способное применить построенный классификатор для изображений с камеры.

В качестве макета приложения был использован пример приложения из официального репозитория Tensorflow. В макете были удалены ненужные части для решения других задач. Далее была добавлена специальная рамка размером 200x300 над выводом с камеры и, соответственно, обрезка изображений с камеры по этой рамке. Это было сделано для уменьшения размера входного изображения в нейронную сеть.

Обрезанное изображение уменьшалось до разрешения 96x96 и подавалась на вход нейронной сети. Выходы нейронной сети преобразовывались из векторного

вида в числовой и отображались вверху экрана телефона. Пример работы приложения можно увидеть на рисунках 20, 21.

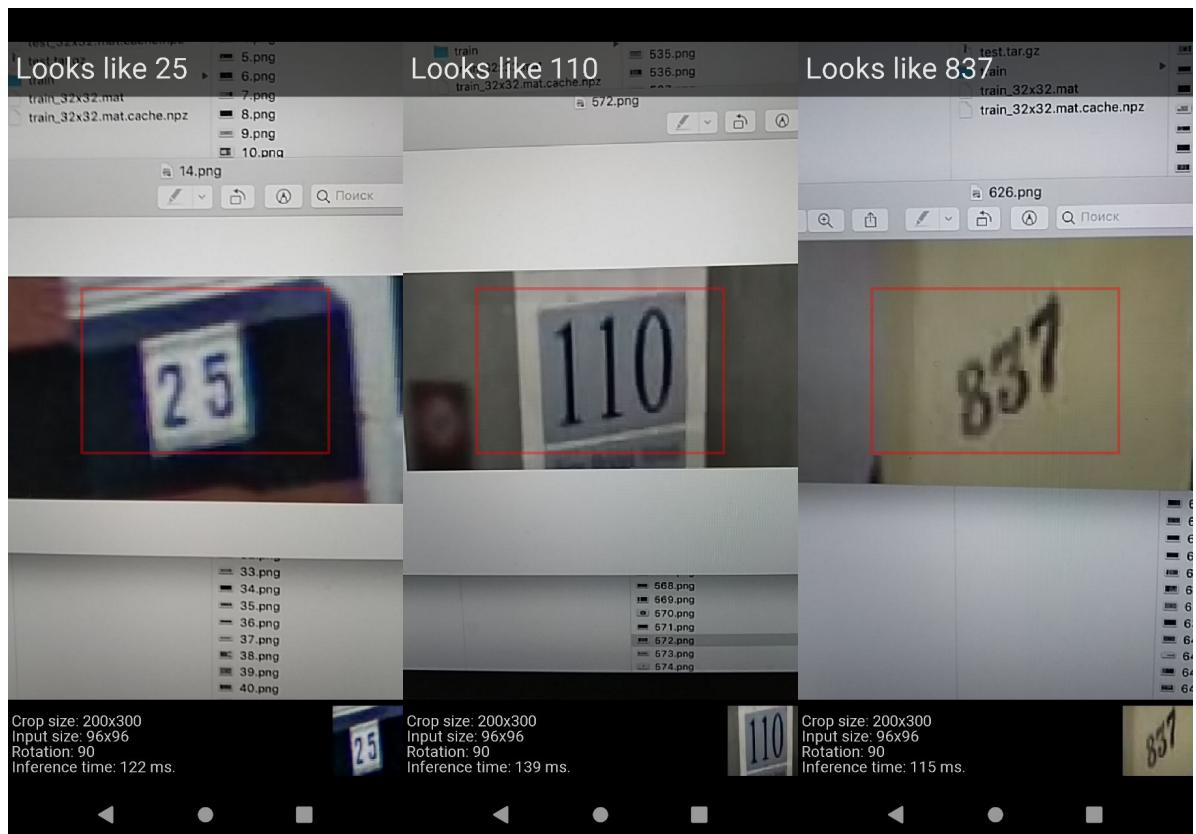


Рис. 20. Примеры работы приложения для номеров домов из архива SVHN.

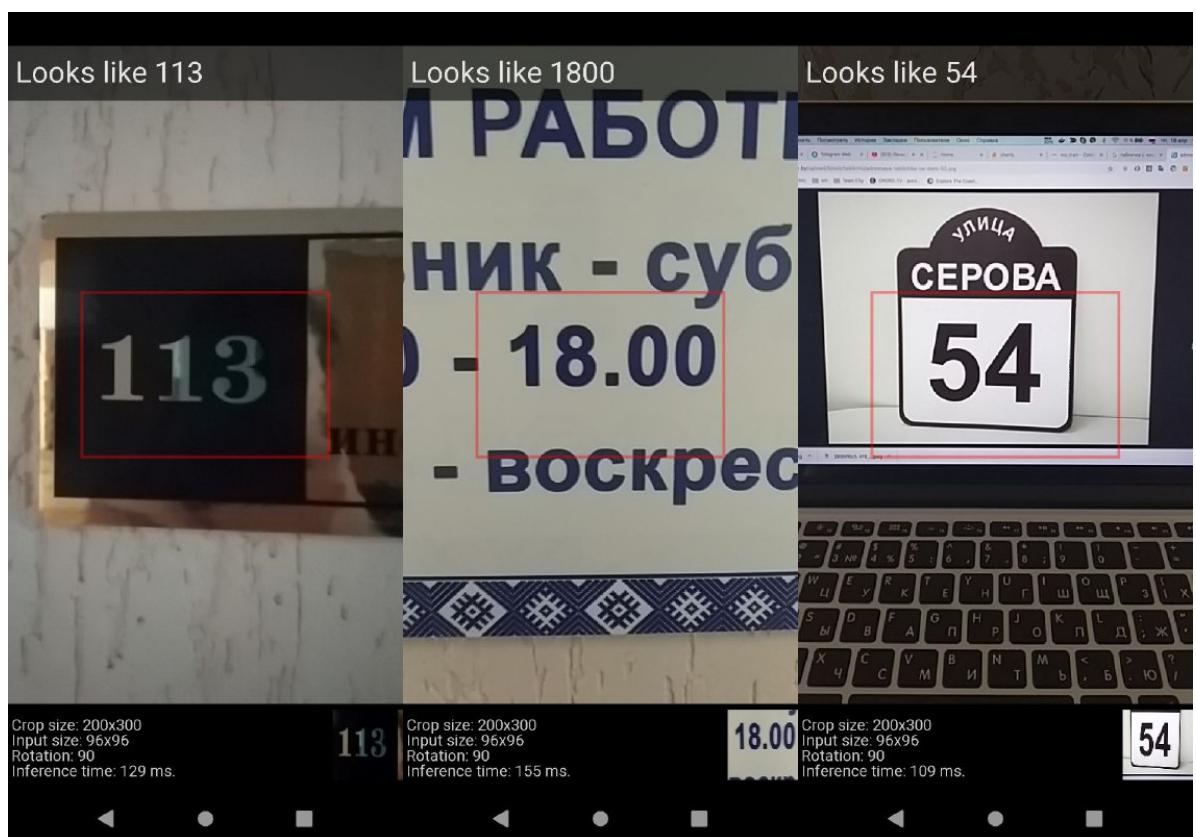


Рис. 21. Примеры работы приложения для номеров из реальной жизни.

Приложение хорошо распознает номера длиной до 3 включительно. Номера длинной четыре распознаются плохо. Номера длиной 5 и 6 практически не распознаются, а если и распознаются, то только отдельные цифры из них. Данных эффект связан с представленностью длин номеров в тренировочной выборке и полностью соответствует их распределению.

Также классификатор оказался чувствителен к центрированию и масштабу изображения номера. Данных эффект связан с тем, что при препроцессинге тренировочной выборки, все номера домов оказались центризованными после обрезки по рамкам и занимали почти весь размер выходного изображения. С данным эффектом можно бороться с помощью аугментации данных: добавлению в тренировочную выборку смещенных изображений по обоим осям.

6. Выход

В результате работы были построены модели для распознавания номеров домов обоих форматов из архива SVHN. Также было реализовано приложение под OS Android, способное распознавать номер дома поданный на вход видеокамеры.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] – SVHN dataset [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://ufldl.stanford.edu/housenumbers/>

[2] - Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://arxiv.org/pdf/1312.6082.pdf>

[3] – Preprocessing SVHN Format 1 [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://github.com/imadmal/i/vhn-format1>

[4] – Tensorflow Android Camera Demo [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://github.com/tensorflow/tensorflow/tree/master/tensorflow/examples/android>

[5] – Mobile Net V2 [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://machinethink.net/blog/mobilenet-v2/>

[6] – Save Keras model as a single Tensorflow .pb file [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://www.dlogy.com/blog/how-to-convert-trained-keras-model-to-tensorflow-and-make-prediction/>