МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
Национальный исследовательский Нижегородский государственный университет
им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

Кафедра математического обеспечения и суперкомпьютерных технологий

Направление подготовки
02.04.02. Фундаментальная информатика и информационные технологии

Направленность образовательной программы
магистерская программа «Компьютерная графика и моделирование живых и технических систем»

Отчёт
по методам глубокого обучения для решения задач компьютерного зрения

на тему:
«Разработка полностью связанных нейронных сетей»

Квалификация (степень)
магистр

Форма обучения
очная

Выполнили: студенты группы 381706-3М
Храмов Илья Валерьевич

_____
Подпись

Реунова Ольга Алексеевна

_____
Подпись

Воеводин Андрей Михайлович

_____
Подпись

Н. Новгород
2018 г.

Содержание

Постановка задачи

Цели

Цель настоящей работы состоит в том, чтобы получить базовые навыки работы с одной из библиотек глубокого обучения (Caffe, Torch, TensorFlow, MXNet или какая-либо другая библиотека на выбор студента) на примере полностью связанных нейронных сетей.

Задачи

Выполнение практической работы предполагает решение следующих задач:

1. Выбор библиотеки для выполнения практических работ курса.
2. Установка выбранной библиотеки на кластере (параметры аутентификации и инструкция по работе с кластером выложена в отдельной задаче в системе redmine).
3. Проверка корректности установки библиотеки. Разработка и запуск тестового примера сети, соответствующей логистической регрессии, для решения задачи классификации рукописных цифр набора данных MNIST (пример разобран в лекционных материалах).
4. Выбор практической задачи компьютерного зрения для выполнения практических работ.
5. Разработка программ/скриптов для подготовки тренировочных и тестовых данных в формате, который обрабатывается выбранной библиотекой.
6. Разработка нескольких архитектур полностью связанных нейронных сетей (варьируются количество слоев и виды функций активации на каждом слое) в формате, который принимается выбранной библиотекой.
7. Обучение разработанных глубоких моделей.
8. Тестирование обученных глубоких моделей.
9. Публикация разработанных программ/скриптов в репозитории на GitHub.
10. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

Тренировочные и тестовые наборы данных

Задача — классификация комиксов. Данные получены из [5]. 86 классов сокращены до 14 классов с целью убрать классы, в которых выборка не репрезентативна. Размер изображений в каждом классе 108*72. Изображения 3 канальные. Для “Сети 5” (см. в разделе “Тестовые конфигурации сетей”) размер изображений — 210*140.

№	Категории	Размер тренировочной выборки	Размер тестовой выборки
1	Aquaman v7	1088	282
2	Batgirl v4	1088	293
3	Batman v2	1509	384
4	Batwing	616	167
5	Batwoman	739	187
6	Catwoman v4	1047	256
7	Green Arrow	1040	256
8	Green Lantern	1301	345
9	Harley Quinn	671	165
10	Nightwing v3	558	147
11	Red Lanterns	767	197
12	Sinestro	619	143
13	Supergirl v6	701	200
14	Wonder Woman	1233	292
		12977	**3314**

Таблица 1. Размер выборки в каждом классе тренировочного и тестового множеств.

Изображения хранятся в формате JPEG.

На вход сети подаются бинарные файлы с расширениями .rec (изображения), .idx (индексы изображений) (Рис. 1).

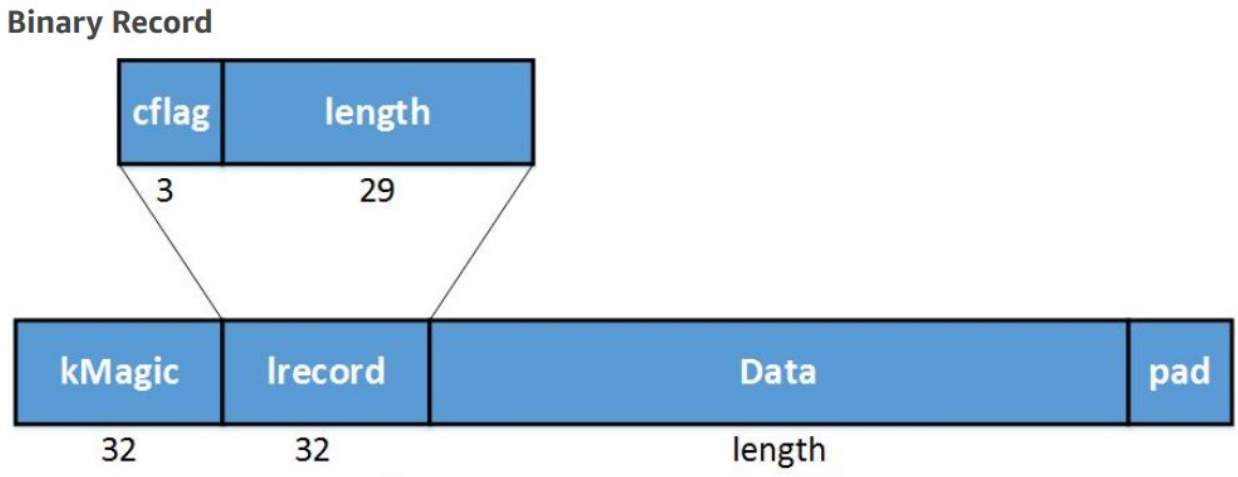

Рисунок 1. Формат хранения в MXNet [8].

- kMagic – начало записи;
- lrecord – длина (length) и продолжительность записи (cflag);
- Data – данные;
- pad – пространство для выравнивания до 4 байт.

Метрика качества решения

В качестве метрики для оценки качества решения задачи [5] выбрана “Точность” (“Accuracy”). В терминологии MXNet — это отношение количества правильно предсказанных сэмплов к общему количеству сэмплов (Рис. 2).

$$\mathbf{accuracy}(y, \hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1} 1(\hat{y}_i == y_i)$$

Рисунок 2. Определение “Accuracy” в MXNet [7].

Более подробную информацию о метриках можно найти в [7].

Разработанные программы/скрипты

- `reduce_dataset.py` - скрипт для вычленения из оригинального набора данных наиболее репрезентативных категорий.
- `prepare_dataset.py` - скрипт для подготовки данных под *mxnet.*
- `lab2.py` - скрипт для обучения полносвязных нейронных сетей

Тестовые конфигурации сетей

1.

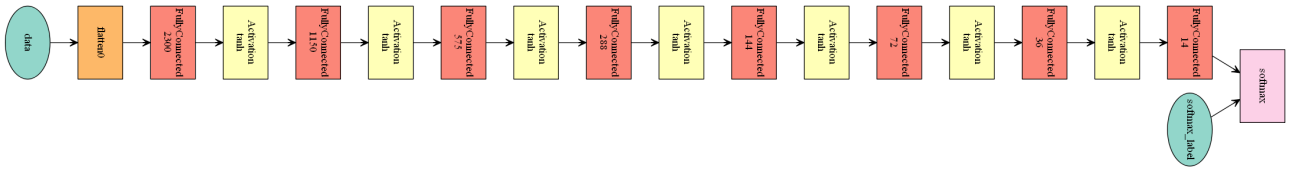

Рисунок 3. Сеть 1.

2.

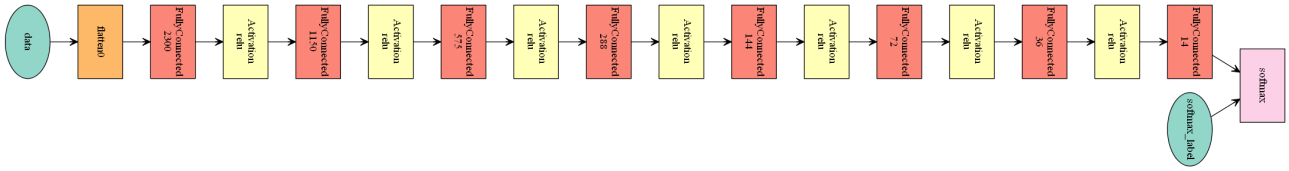

Рисунок 4. Сеть 2.

3.

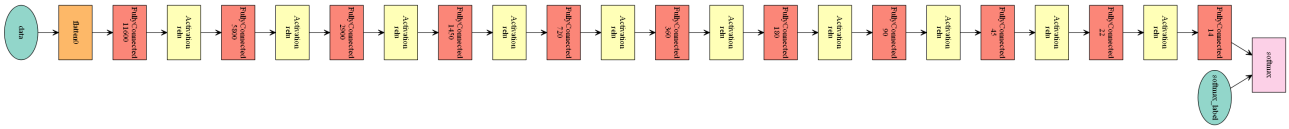

Рисунок 5. Сеть 3.

4.

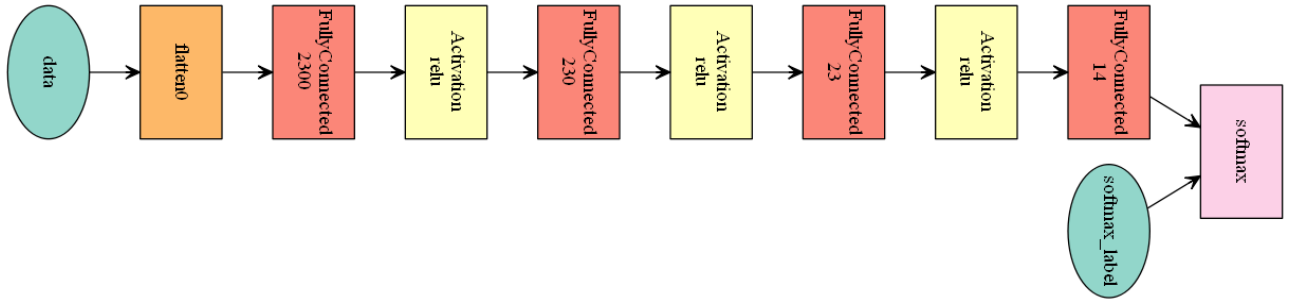

Рисунок 6. Сеть 4.

5.

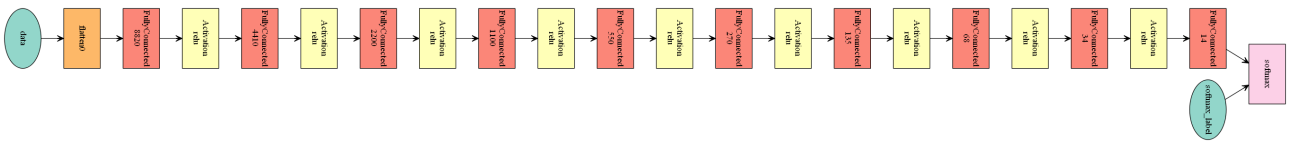

Рисунок 7. Сеть 5.

Результаты

В таблице 2 приведены конфигурация системы и программное обеспечение, с помощью которых проводилось обучение и тестирование построенных моделей.

Параметры	**Версия**
Операционная система	Windows10
GPU	NVIDIA GeForce GTX 1080 (частота процессора — OC Mode – GPU Boost Clock : 1835 MHz , GPU Base Clock : 1695 MHz Gaming Mode (Default) - GPU Boost Clock : 1809 MHz , GPU Base Clock : 1670 MHz; шина передачи данных – PCI Express 3.0; видеопамять - GDDR5X 8GB; количество ядер — 2560; частота памяти - 10010 MHz; интерфейс памяти — 256-bit;)
CUDA	9.2
Python	3.7.1
MXNet	1.3.0

Таблица 2. Конфигурация системы.

В таблице 3 приведены параметры обучения.

Параметры обучения	
Оптимизатор	SGD (стохастический градиентный спуск)
Скорость обучения	0.0001
Количество эпох	60
Размер batch	10

Таблица 3. Параметры обучения.

	Наименование сети				
	Сеть 1	Сеть2	Сеть 3	Сеть 4	Сеть 5
Среднее время обучения за одну эпоху, с	10,53	10,49	52,13	8,18	104,21
Качество решения на тренировочном наборе (Accuracy), %	13,85	92,92	92,71	10,21	87,29
Качество решения на тестовом наборе (Accuracy), %	14,2	18,1	19,61	11,57	21,69
Номер эпохи с достигнутым максимальным качеством решения на тренировочном наборе	44	51	51	25	59
Максимальное качество решения на тренировочном наборе (Accuracy), %	17,4	94,54	94,64	13,3	89,79
Номер эпохи с достигнутым максимальным качеством решения на тестовом наборе	30	43	51	2	46
Максимальное качество решения на тестовом наборе (Accuracy), %	14,95	20,6	20,03	11,63	22,29

Таблица 4. Результаты экспериментов. Конфигурация сетей приведена в “Тестовые конфигурации сетей”.

Анализ результатов

Из представленных результатов можно сделать выводы:

1. Выборка — нерепрезентативная. Малое количество изображений на один класс, как тренировочной, так и тестовой выборок. Некоторые изображения в тестовой и тренировочной выборках слишком отличаются (например, тренировочная выборка — нарисован Бэтман на черном фоне, тестовая выборка — нарисован Бэтман на светлом фоне). Герои одних комиксов могут встречаться в других комиксах (например, Harley Quinn в Batman и наоборот).
2. Плохой выбор скорости обучения. Если данный параметр слишком большой, то точность будет приблизительно 11% (измерялось для разных конфигураций сетей), то есть мы будем бесконечно «прыгать» через точку минимума. Если мы берем параметр слишком маленьким, то мы “застреваем” в локальном минимуме.
3. Начальная обработка данных. Как видно из таблицы 4, для конкретного размера изображения необходимо настраивать количество нейронов так, чтобы избежать “узкого горлышка”.
4. Надо понимать, что для таких задач лучше использовать сверточные или другие нейронные сети.

Литература

1. MNIST dataset [http://yann.lecun.com/exdb/mnist].
2. OpenCV [http://opencv.org].
3. Материалы Летней межвузовской школы 2016 [https://github.com/itseez-academy/itseez-ss-2016- theory], [https://github.com/itseez-academy/itseez-ss-2016-practice].
4. Лекции по глубокому обучению: https://sites.google.com/site/kustikovavalentina/studentam/kurs-glubokoe-obucenie, 2018.
5. Исходные данные — https://www.kaggle.com/cenkbircanoglu/comic-books-classification: kaggle datasets download -d cenkbircanoglu/comic-books-classification.
6. Документация MXNet — http://mxnet.incubator.apache.org/test/tutorials/.
7. Метрики в MXNet – https://mxnet.incubator.apache.org/api/python/metric/metric.html.
8. Формат хранения данных в MXNet: https://mxnet.incubator.apache.org/architecture/note_data_loading.html.
9. Репозиторий исходных кодов: https://github.com/okondratieva/DeepLearning.