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

Институт информационных технологий, математики и механики

Направление подготовки: «Прикладная математика и информатика»
Магистерская программа: «Вычислительные методы и суперкомпьютерные технологии»

Образовательный курс «Глубокое обучение»

ОТЧЕТ

по лабораторной работе №3

Разработка сверточных нейронных сетей

Выполнили:
студенты группы 381703-3м
Гладкова Татьяна
Крутоборежская Ирина
Крюкова Полина
Подчищаева Мария

Нижний Новгород
2018

Содержание

Цели

Цель настоящей работы состоит в том, чтобы построить архитектуру сверточной нейронной сети, которая позволяет решать практическую задачу с высокими показателями качества.

Задачи

Выполнение практической работы предполагает решение ***следующих задач***:

1. Разработка нескольких архитектур сверточных нейронных сетей (варьируются количество слоев и виды функций активации на каждом слое) в формате, который принимается выбранной библиотекой глубокого обучения.

2. Обучение разработанных глубоких моделей.

3. Тестирование обученных глубоких моделей.

4. Публикация разработанных программ/скриптов в репозитории на GitHub.

5. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

Решаемая задача

Была выбрана задача бинарной классификации: «кошки» - «собаки». Были использованы картинки из наборов данных https://www.kaggle.com/tongpython/cat-and-dog и https://www.kaggle.com/c/dogs-vs-cats/data. Получившийся набор состоит из 35029 изображений.

Рис. 1 Пример изображения из класса «кошки»

Рис. 2 Пример изображения из класса «собаки»

С помощью скрипта на python данные были преобразованы к размеру 128×128. С помощью скрипта im2rec.py, который входит в библиотеку MXNet, изображения были сконвертированы в формат .rec.

Метрика качества решения задачи

В качестве метрики точности решения используется отношение угаданных животных ко всем в тестовой выборке:

$$Accuracy = \frac{CorrectAnswersCount}{ImagesCount}$$

Тренировочные и тестовые наборы данных

В качестве тренировочной выборки используем тренировочную выборку первого и второго наборов данных, всего 16500 изображений котов и 16505 изображений собак. В качестве тестовой выборки используем тестовую выборку только из первого набора данных, т.к. во втором наборе данных тестовая выборка не разбита на изображения котов и собак. Всего в тестовой выборке 2042 изображения, котов и собак поровну.

Конфигурации нейронных сетей

В данной работе были рассмотрены четыре конфигурации сверточных нейронных сетей с одним и двумя сверточными слоями. Ядра сверточных слоев выбраны следующих размеров:

- 2x2
- 3x3

В каждом сверточном слое к картинке может применяться по несколько фильтров. Значение фильтров выбирались из следующих:

- 500
- 1000

Так же в сверточных слоях фильтр шел по картинке с некоторым шагом по пикселям. Этот шаг называется stride. И использовались следующие его значения:

- (2, 2)
- (1, 1)

После каждой свертки применялась функция активации:

- $tanh, f = \frac{e^{s} - e^{-s}}{e^{v} + e^{-s}}$
- $relu, fmax(x, 0)$

После активации используется операция пространственного объединения (pooling):

- Max – pooling $f = max(x_i)$

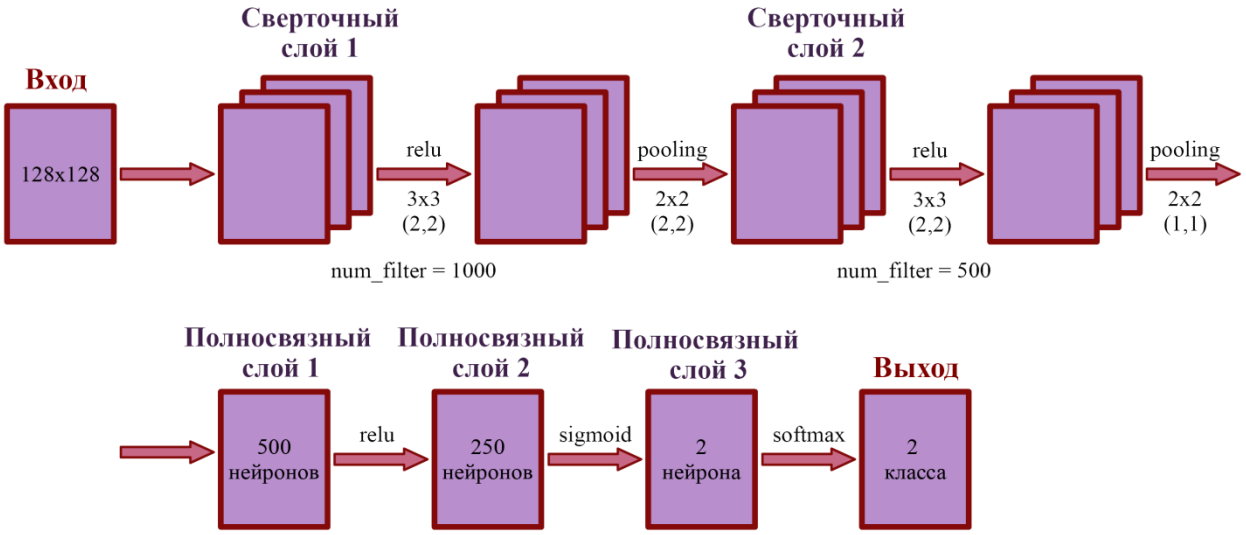
Активационная функция на полносвязных слоях выбирается из следующих:

- $tanh, f = \frac{e^{s} - e^{-s}}{e^{v} + e^{-s}}$
- $sigmoid, f \frac{1}{e^{v} + e^{-s}}$
- $relu, fmax(x, 0)$

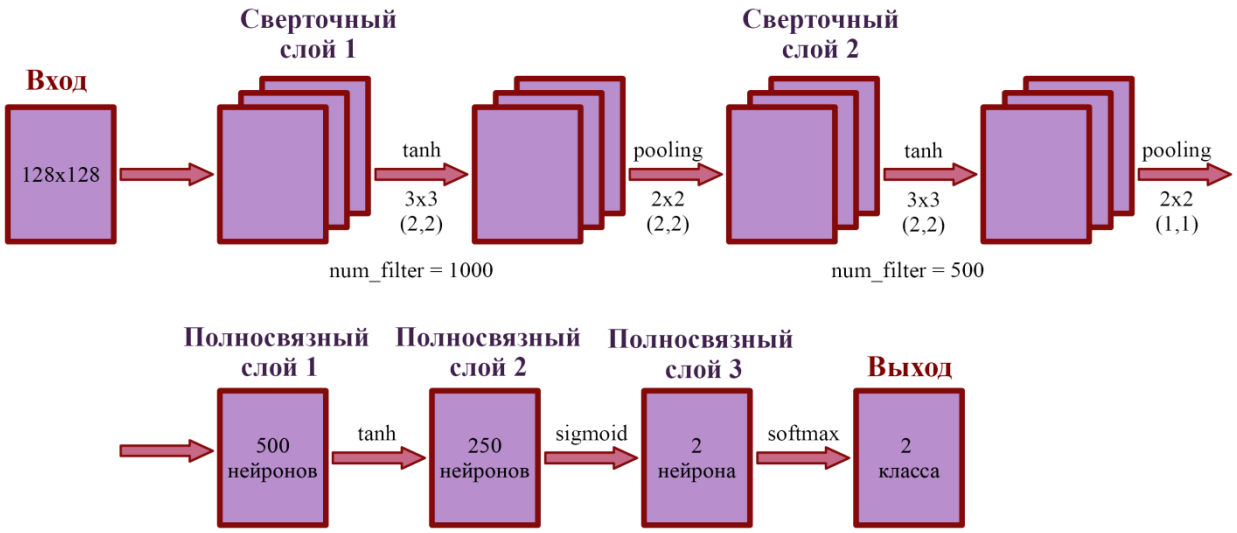
На выходном слое:

- $softmax, f \frac{e^{s_j}}{\sum_{j=1}^{n} e^{s_j}}$

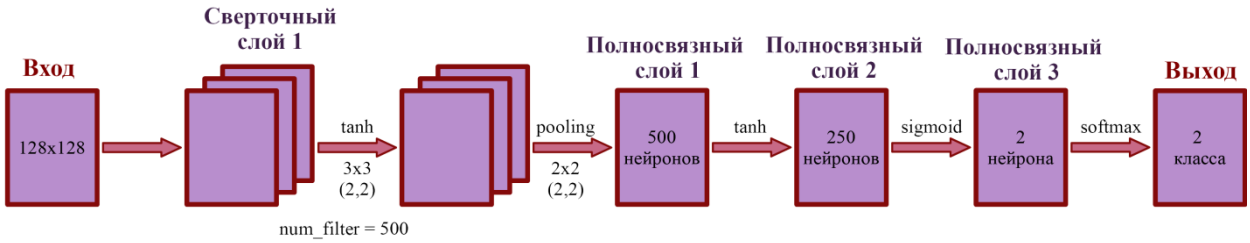
Конфигурация №1



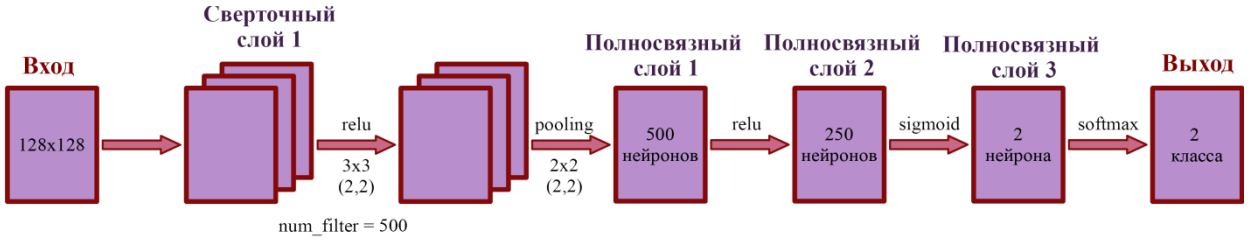
Конфигурация №2



Конфигурация №3



Конфигурация №4



Разработанные программы/скрипты

В директории расположены четыре конфигурации построенных сверточных нейронных сетей. Соответствия построенных конфигураций и конфигураций в директории:

- CNN_config_1.py – первая нейронная сеть с двумя сверточными слоями и с тремя полносвязными слоями, функции активации relu.
- CNN_config_2.py – вторая нейронная сеть с двумя сверточными слоями и с тремяполносвязными слоями, функции активации tanh.
- CNN_config_2.py – третья нейронная сеть с одним сверточным слоем и с тремяполносвязными слоями, функции активации tanh.
- CNN_config_4.py – четвертая нейронная сеть с одним сверточным слоем и с тремяполносвязными слоями, функция активации relu.

Результаты экспериментов

В работе рассмотрены 4 конфигурации.

Тестовая инфраструктура
Вычисления производились на машине со следующими характеристиками:

- ОС: Windows 10
- Процессор: Intel(R) Core(TM) i7-6700k CPU @ 4.00GHz 4.01 GHz
- Установленная Память (ОЗУ): 16,0 ГБ
- Тип системы: 64 – разрядная операционная система, процессор x64
- Видеокарта: NVIDIA GeForce GTX 1070

Параметры обучения:

- количество эпох – 10,
- скорость обучения – 0.001.

№	**Число сверточных слоев**	**Число полносвязных слоев**	**Функции активации**	**Результат**		
				Точность на тренировочном множестве	Точность на тестовом множестве	Время обучения, с
1	2	2	relu-relu-sigmoid	0.94	0.94	1310.58
2	2	2	tanh-tanh-tanh-sigmoid	0.74	0.75	1311.33
3	1	2	tanh-tanh-sigmoid	0.63	0.69	1722.54
4	1	2	relu-relu-sigmoid	0.99	**0.98**	1716.83

Выводы

Наихудший результат был получен с помощью нейронной сети конфигурации №3, так как в этой сети в качестве функции активации была использована функция tanh (сигмоида), которая (как было показано в л/р №2) обычно оказывается хуже ReLU.

Наилучший результат был получен на нейронной сети с конфигурацией №4. Сверточные нейронные сети дали лучший результат по сравнению с полносвязными нейронными сетями. Максимальная точность в сверточных нейронных сетях – 0.99, а в полностью связанных – 0.83.