ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
”ВЫСШАЯ ШКОЛА ЭКОНОМИКИ”»

**Московский институт электроники и математики**

Юткин Дмитрий Игоревич, группа БИВ-141

# ПРИМЕНЕНИЕ ГЛУБОКИХ СВЁРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ РЕШЕНИЯ ЗАДАЧИ РАСПОЗНАВАНИЯ ОБРАЗОВ

Междисциплинарная курсовая работа
по направлению 09.03.01.62 Информатика и вычислительная техника
студента образовательной программы бакалавриата
«Информатика и вычислительная техника»

Студент _____ Д.И. Юткин

Научный руководитель
Старший преподаватель
Д.В. Пантюхин

_____

Москва 2016 г.

## Аннотация

Работа посвящена решению задачи распознавания образов с использованием глубоких свёрточных нейронных сетей. В результате исследования было обучено несколько нейронных сетей различной глубины (от 13 до 16 слоёв), часть из которых обучалась на предварительно обработанных данных. Максимальная точность одиночной модели составила 93,33%, объединение нейронных сетей в ансамбль позволило увеличилась точность распознавания до 94,15%, что сравнимо с точностью человека. Все эксперименты проводились на наборе изображений CIFAR-10, с использованием графических процессоров и фреймворка для глубокого обучения Caffe.

## Оглавление

**1 Введение** . . . . . 4

**2 Основная часть** . . . . . 6

2.1 Датасет CIFAR-10 . . . . . 6

2.2 Фреймворк Caffe . . . . . 7

2.3 Предварительная обработка данных . . . . . 7

2.4 Обучение нейронных сетей . . . . . 9

2.4.1 Модель I . . . . . 9

2.4.2 Модель II . . . . . 10

2.5 Анализ результатов одиночных моделей . . . . . 10

2.6 Объединение нейронных сетей в ансамбль . . . . . 10

**Список литературы** . . . . . 11

**Приложение** . . . . . 12

# 1 Введение

В последние годы компьютерное зрение является одной из самых активно развивающихся областей искусственного интеллекта. Подобный интерес к области обусловлен недавними успехами в решении такой задачи, как обучение компьютера воспроизводить зрительные способности человека, например, распознавать и классифицировать образы.

До недавнего времени распознавание объектов осуществлялось с помощью алгоритмов, для которых вручную приходилось проектировать признаки (feature engineering). Основной недостаток такого подхода — невозможность находить в изображении средние и многоуровневые абстракции, такие как части объекта или пересечения различных краёв. Кроме того, признаки созданные вручную для одного типа изображений зачастую не были применимы к другим. Однако, недавние разработки в области машинного обучения, известные как «глубокое обучение» (deep learning), показали, как вычислительные модели, состоящие из множества слоёв, могут автоматически изучать иерархии признаков прямо из набора данных.

На сегодняшний день глубокое обучение развилось и укрепилось в отдельную ветвь машинного обучения, алгоритмы которой способствовали значительному улучшению результатов в различных задачах, таких как обработка естественного языка, распознавание визуальных образов и др. Благодаря технологиям глубокого обучения сегодня мы можем искать похожие фотографии в Google Photos, или восхищаться беспилотными автомобилями, которые управляются искусственным интеллектом.

Свёрточные нейронные сети (СНН) — одна из главных причин прорыва в компьютерном зрении. Изначально представленные в 1980 году Кунихикой Фукусимой как NeoCognitron [1], а затем улучшенные в 1998 году Яном Лекуном до LeNet-5 [2], СНН получили славу благодаря впечатляющему успеху в распознавании рукописных цифр. Для следующего прорыва в компьютерном зрении потребовалось чуть больше двадцати лет. С увеличением производительности графических процессоров (GPU), стало возможным обучение глубоких нейронных сетей. В 2012 году глубокая СНН победила в мировом соревновании ImageNet Large-scale Visual Recognition Challenge (ILSVRC), значительно превзойдя предыдущие результаты, достигнутые алгоритмами полагающимися на ручную генерацию признаков [3].

Таким образом, темой данного исследования является разработка системы распознавания объектов (object recognition), основанной на глубоких свёрточных нейронных сетях. Задача решалась с помощью фреймворка для глубокого обучения Caffe на наборе данных CIFAR-10. Основные цели исследования заключались в следующем:

а) Изучить и применить на практике свёрточные нейронные сети в решении задачи распознавания объектов;
б) Изучить современные технологии машинного обучения, анализа данных и глубокого обучения.

Для достижения поставленных целей были сформулированы следующие задачи:

а) Провести обзор статей и литературы, посвящённых свёрточным нейронным сетям и решению задачи распознавания образов;
б) Выбрать и изучить программное обеспечение для глубокого обучения;
в) Подобрать набор данных, на котором будут проводиться эксперименты;
г) Спроектировать и обучить нейронные сети различных архитектур;
д) Оценить и сравнить точности полученных нейронных сетей;
е) Объединить отдельно обученные модели в ансамбль, с целью увеличения точности распознавания.

Все поставленные задачи выполнены, цели исследования достигнуты. Результаты каждой из задач будут описаны далее, в основной части работы.

## 2 Основная часть

### 2.1 Датасет CIFAR-10

Набор данных, на котором проводились исследования, состоит из 60 000 цветных изображений, размера 32×32 пикселя. Каждое изображение принадлежит одному из 10 классов, что соответствует 600 изображениям на класс. Под обучение отводится 50 000 изображений. Остальные 10 000 используются для тестирования. Объекты в классах сильно варьируются, например, класс «птица» содержит различные виды птиц, как большие так и маленькие. Кроме того, объекты классов представлены в различных позах и под различными углами. Особенно это проявляется среди собак и котов, которые изображены не только в различных позах, но иногда и частично, например, изображена только голова животного.

Датасет CIFAR-10 [4] был выбран для проведения исследований благодаря своему относительно небольшому размеру, который позволяет обучать глубокие нейронные сети используя GPU с памятью меньше 8 Gb, например, в данной работе использовалась NVIDIA Grid K520 с 4 Gb видеопамяти.

На момент написания работы лучший результат (state-of-the-art) на CIFAR-10 составляет 96,53% [5]. Точность распознавания человека равна $\approx 94\%$.[1]

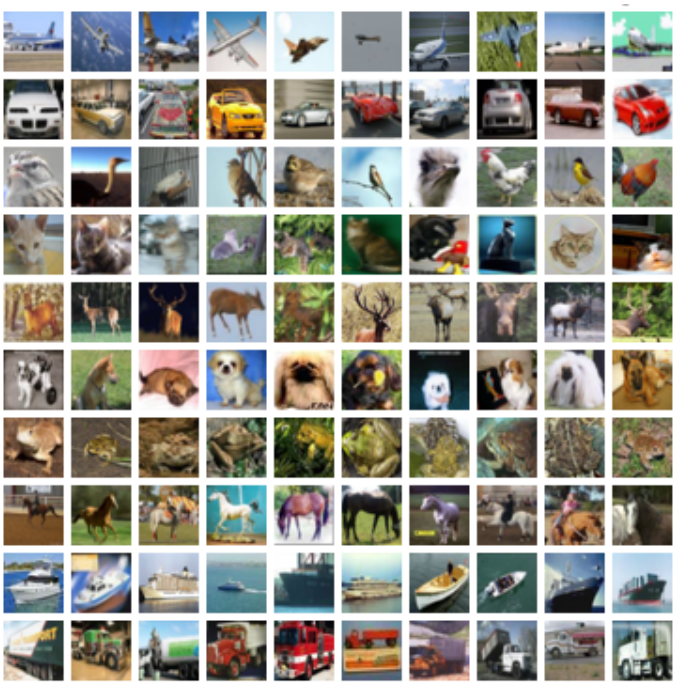


Рисунок 1 — 10 случайных изображений из каждого класса CIFAR-10

[1] http://karpathy.github.io/2011/04/27/manually-classifying-cifar10/

## 2.2 Фреймворк Caffe

Обучение нейронных сетей проводилось с использованием фреймворка для глубокого обучения Caffe [6]. Изначально, фреймворк разрабатывался командой BVLC (Berkeley Vision and Learning Center), но постепенно перерос в большой open-source проект.[2] На данный момент вклад в развитие Caffe внесли почти 200 разработчиков и более 10 000 человек оценили проект на Github.

Данный фреймворк был выбран для решения задачи по нескольким причинам:

а) Простота определения моделей и методов оптимизации. Топологии нейронных сетей и алгоритмы оптимизации удобно и эффективно определяются в специальных конфигурационных файлах типа Google Protocol Buffers.[3] Caffe поддерживает топологии сетей в форме любых ациклических графов.

б) Модульность. Caffe позволяет легко изменять архитектуру сети под новые форматы входных данных. Кроме того, в наличии имеется много слоёв и функций потерь.

в) Скорость вычислений и эффективное использование ресурсов. Caffe заранее выделяет ровно столько памяти сколько нужно для нейронной сети. Операций линейной алгебры такие как умножение, сложение и свёртка выполняются на CPU с помощью BLAS (Basic Linear Algebra Subroutines). На GPU за эти операции отвечают библиотеки cuBLAS и cuDNN [7]

г) CLI (command line interface) и интерфейсы для Python и Matlab.

Caffe написан на языках C++, Cuda, Python. Для хранения больших объёмов данных используются базы данных LMDB[4] и LevelDB[5]. Обучение нейронных сетей может выполняться как на CPU, так и на нескольких GPU одновременно. Фреймворк доступен для установки на Linux, Windows и OS X.

## 2.3 Предварительная обработка данных

Необработанные изображения содержат излишнюю информацию, так как смежные пиксели имеют высокую корреляцию. Поэтому прежде чем подавать

[2] https://github.com/BVLC/caffe
[3] https://developers.google.com/protocol-buffers/
[4] http://symas.com/mdb/
[5] https://github.com/google/leveldb

изображения на вход нейронной сети, они были обработаны в два этапа. В начале, была произведена глобальная нормализация контраста (global contrast normalization) изображения:

$$\widehat{X} = \frac{X - \overline{X}}{\sigma},$$

где $X$ — исходное изображение, $\overline{X}$ — среднее значение, $\sigma$ — стандартное отклонение.

Затем изображения были линейно трансформированы с помощью алгоритма «ZCA whitening» [4]. Цель данного алгоритма сделать так, чтобы входные изображения слабо коррелировали друг с другом.

Алгоритм «ZCA whitening»:

```
1 cov = np.dot(X.T, X) / X.shape[0] # Вычисляем ковариационную матрицу
2 U,S,V = np.linalg.svd(cov) # Находим сигнулярное разложение ковариацонной матрицы
3 Xrot = np.dot(X, U) # Поворачиваем входные данные
4 Xwhite = Xrot / np.sqrt(S + α) # Делим на собственные числа
```

Рисунок 2 — Изображения после применения «ZCA whitening» с параметром $\alpha = 0{,}1$

Предобработка с помощью «ZCA whitening» уменьшила ошибку в среднем на $1{,}5\%$. Кроме того, каждое изображение было увеличено до $40\times40$ пикселей, для того чтобы в процессе обучения вырезать из изображения случайный патч размером $32\times32$. Такой приём позволяет снизить эффект переобучения модели и повысить конечную точность распознавания.

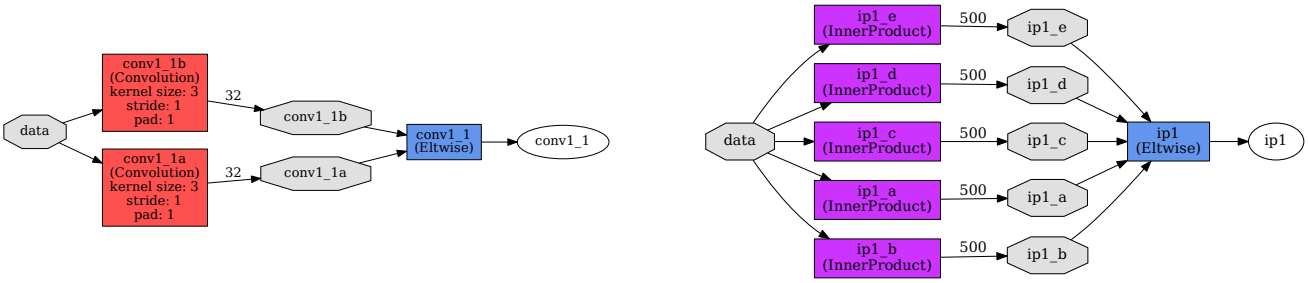
## 2.4 Обучение нейронных сетей

Походу исследования были проверены различные гипотезы, архитектуры и эвристики так или иначе влияющие на конечные точности моделей. В результате различных экспериментов были отобраны семь нейронных сетей, показавшие наилучший результат на тестировании. Далее будут описаны каждая из отобранных сетей, их свойства и особенности обучения.

### 2.4.1 Модель I

Модель I имеет самую низкую ошибку в предсказаниях среди всех обученных моделей (6,7%). СНН состоит из 18 обучающихся слоёв. Всего в нейронной сети ≈ 2,7 млн. параметров. Архитектура сети отражена в таблице 1.

Maxout в качестве функции активации является её отличительной особенностью. На рисунке 3а показано как maxout соединён с выходами свёрточных слоёв. На рисунке 3б можно видеть соединение выходов полносвязного слоя и активации. Также, после каждого «пулинг» (pooling) слоя расположен «дропаут» (dropout) с параметром 0,2, который помогает снизить эффект переобучения сети.




а) б)

Рисунок 3 — Пример соединения maxout активаций в Модели I

Нейронная сеть обучалась 90 000 итераций методом стохастического градиентного спуска с моментом (stochastic gradient descent with momentum), с начальным learning rate (lr) 0,01 и моментом 0.9. Кроме того, использовалась L2 регуляризация с коэффициентом 0,0005. В процессе обучения, начиная с 40 000 итерации, lr уменьшался в десять раз каждые 20 000 итераций. Таким образом, к концу обучения lr достиг значения $10^{-5}$. Mini batch состоял из 256 изображений. Графики обучения представлены на рисунке 4.

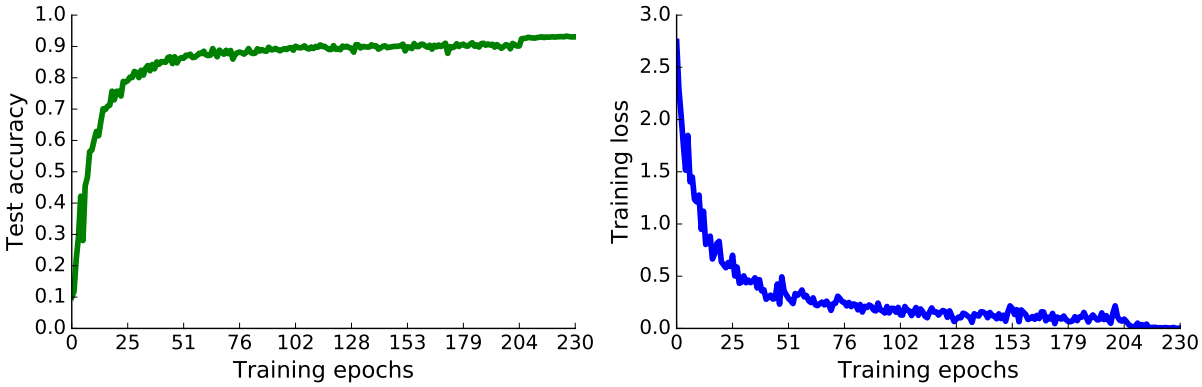



Рисунок 4 — Графики изменения точности и ошибки в процессе обучения Модели I

### 2.4.2 Модель II

Модель II имеет в 4,5 раза меньше параметров, чем Модель I. Их число составляет $\approx 600$ тысяч. Данная нейронная сеть имеет классическую архитектуру, — чередование свёрточных и «пулинг» слоёв, в качестве функции активации используется Leaky ReLU[6] с параметром $\alpha = 0{,}01$.

## 2.5 Анализ результатов одиночных моделей

## 2.6 Объединение нейронных сетей в ансамбль

[6] $f(x) = \max(x, \alpha x)$

## Список литературы


1. *Fukushima Kunihiko*. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position // *Biological Cybernetics*. — Vol. 36, no. 4. — Pp. 193–202.

2. Gradient-Based Learning Applied to Document Recognition / Y. LeCun, L. Bottou, Y. Bengio, P. Haffner // *Proceedings of the IEEE*. — 1998. — November. — Vol. 86, no. 11. — Pp. 2278–2324.

3. *Krizhevsky Alex, Sutskever Ilya, Hinton Geoffrey E.* ImageNet Classification with Deep Convolutional Neural Networks // Advances in Neural Information Processing Systems 25 / Ed. by F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger. — Curran Associates, Inc., 2012. — Pp. 1097–1105.

4. *Krizhevsky Alex*. Learning multiple layers of features from tiny images. — 2009.

5. *Graham B.* Fractional Max-Pooling // *ArXiv e-prints*. — 2014. — dec. http://adsabs.harvard.edu/abs/2014arXiv1412.6071G.

6. Caffe: Convolutional Architecture for Fast Feature Embedding / Yangqing Jia, Evan Shelhamer, Jeff Donahue et al. // *arXiv preprint arXiv:1408.5093*. — 2014.

7. cuDNN: Efficient Primitives for Deep Learning / Sharan Chetlur, Cliff Woolley, Philippe Vandermersch et al. // *CoRR*. — 2014. — Vol. abs/1410.0759. http://arxiv.org/abs/1410.0759.

# Приложение

Таблица 1: Архитектуры обученных моделей

| Model I (93,3%) | Model II (89,7%) | Model III (91,0%) | Model IV (87,3%) | Model V (89,3%) | Model VI (92,0%) | Model VII (92,3%) |
|---|---|---|---|---|---|---|
| conv1_1 3x3x32 | conv1_1 3x3x32 | conv1_1 3x3x64 | conv1 4x4x32 | conv1 3x3x32 | conv1_1 3x3x64 | conv1_1 3x3x32 |
| conv1_2 3x3x32 | conv1_2 3x3x32 | conv1_2 3x3x64 | conv2 4x4x64 | conv2 3x3x32 | conv1_2 3x3x64 | conv1_2 3x3x32 |
| conv1_3 3x3x32 | conv1_3 3x3x32 | conv1_3 3x3x64 | conv3 4x4x96 | conv3 3x3x64 | conv1_3 3x3x64 | conv1_3 3x3x32 |
| conv1_4 3x3x48 | pool 2x2 | pool 2x2 | conv4 4x4x128 | conv4 3x3x64 | conv1_4 3x3x64 | conv1_4 3x3x48 |
| conv1_5 3x3x48 | conv2_1 3x3x64 | conv2_1 3x3x128 | conv5 4x4x160 | conv5 3x3x128 | conv1_5 3x3x64 | conv1_5 3x3x48 |
| pool 2x2 | conv2_2 3x3x64 | conv2_2 3x3x128 | conv6 4x4x192 | conv6 3x3x128 | pool 2x2 | pool 2x2 |
| conv2_1 3x3x80 | conv2_3 3x3x64 | conv2_3 3x3x128 | conv7 4x4x224 | conv7 3x3x256 | conv2_1 3x3x128 | conv2_1 3x3x80 |
| conv2_2 3x3x80 | conv2_4 3x3x64 | conv2_4 3x3x128 | conv8 4x4x256 | conv8 3x3x256 | conv2_2 3x3x128 | conv2_2 3x3x80 |
| conv2_3 3x3x80 | pool 2x2 | pool 2x2 | conv9 4x4x228 | conv9 3x3x256 | conv2_3 3x3x128 | conv2_3 3x3x80 |
| conv2_4 3x3x80 | conv3_1 3x3x128 | conv3_1 3x3x256 | fc 256 | conv10 3x3x256 | conv2_4 3x3x128 | conv2_4 3x3x80 |
| conv2_5 3x3x80 | conv3_2 3x3x128 | conv3_2 3x3x256 | softmax 10 | conv11 3x3x256 | conv2_5 3x3x128 | conv2_5 3x3x80 |
| conv2_6 3x3x80 | conv3_3 3x3x128 | conv3_3 3x3x256 | | conv12 3x3x512 | conv2_6 3x3x128 | conv2_6 3x3x80 |
| pool 2x2 | conv3_4 3x3x128 | conv3_4 3x3x256 | | conv13 3x3x512 | pool 2x2 | pool 2x2 |
| conv3_1 3x3x128 | pool global | pool 2x2 | | conv14 3x3x512 | conv3_1 3x3x256 | conv3_1 3x3x128 |
| conv3_2 3x3x128 | fc 256 | fc 256 | | fc 512 | conv3_2 3x3x256 | conv3_2 3x3x128 |
| conv3_3 3x3x128 | softmax 10 | softmax 10 | | softmax 10 | conv3_3 3x3x256 | conv3_3 3x3x128 |
| conv3_4 3x3x128 | | | | | conv3_4 3x3x256 | conv3_4 3x3x128 |
| conv3_5 3x3x128 | | | | | conv3_5 3x3x256 | conv3_5 3x3x128 |
| conv3_6 3x3x128 | | | | | conv3_6 3x3x256 | conv3_6 3x3x128 |
| pool global | | | | | pool global | pool global |
| fc 500 | | | | | fc 512 | fc 512 |
| softmax 10 | | | | | softmax 10 | softmax 10 |