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 9 до 18 слоёв), часть из которых обучалась на предварительно обработанных данных. Максимальная точность одиночной модели составила 93,4%, 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

В последние годы компьютерное зрение является одной из самых активно развивающихся областей искусственного интеллекта. Подобный интерес к области обусловлен недавними успехами в обучении 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) [1], показали, как вычислительные модели, состоящие из множества слоёв, могут автоматически изучать иерархии признаков 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.

Свёрточные нейронные сети (СНН) — одна из главных причин прорыва в компьютерном зрении. Изначально представленные в 1980 году Кунихикой Фукусимой как NeoCognitron [2], а затем улучшенные в 1998 году Яном Лекуном до LeNet-5 [3], 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 [4].

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 [5] 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 96,53% [6]. Точность распознавания человека ≈ 94%.[1]

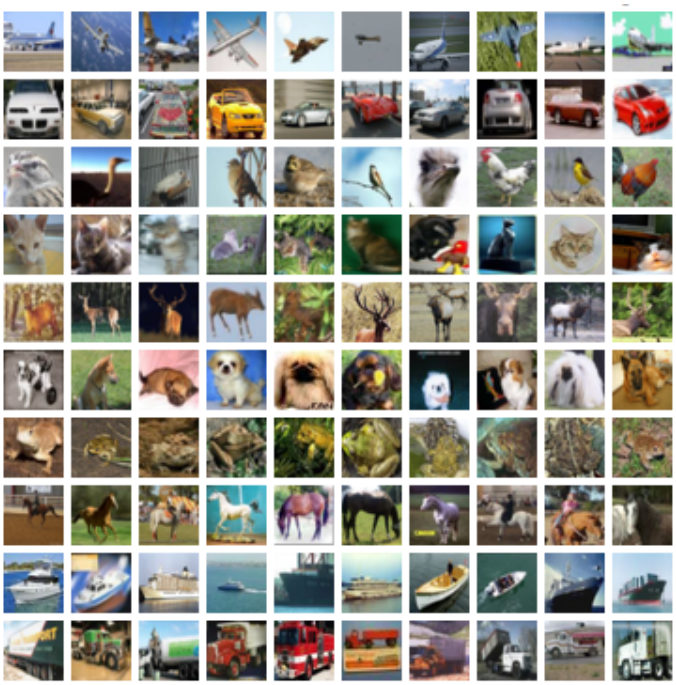


Рисунок 1 — Датасет CIFAR-10

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

2.2 Фреймворк Caffe

Свёрточные нейронные сети обучались с помощью фреймворка для глубокого обучения Caffe [7]. 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 (Caffe поддерживает топологии сетей в форме любых ациклических графов.) и алгоритмы оптимизации удобно и эффективно определяются в специальных конфигурационных файлах типа Google Protocol Buffers[3].

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 [8]

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, σ — стандартное отклонение.

Затем изображения были линейно трансформированы с помощью алгоритма ZCA whitening [5]. 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 40×40 пикселей, для того чтобы в процессе обучения вырезать из изображения случайный патч размером 32×32. Такой приём позволяет снизить эффект переобучения модели и повысить конечную точность распознавания.

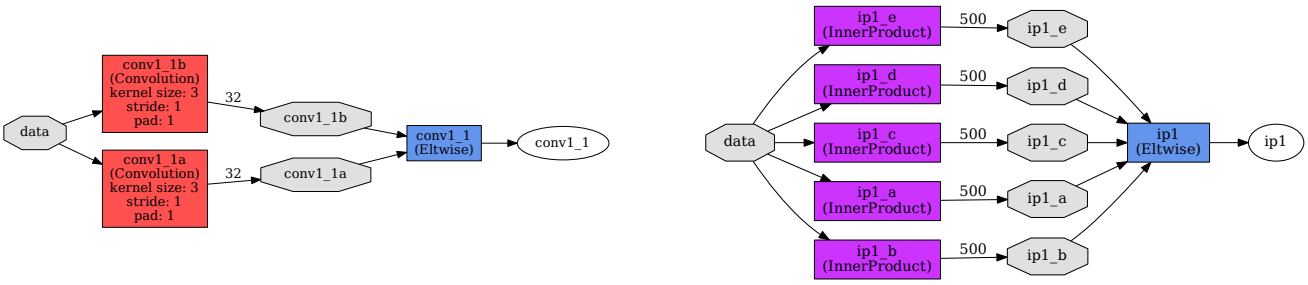
2.4 Обучение нейронных сетей

Во время 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. Топологии сетей отражены в таблице 1, графики обучения на рисунке 7. Далее будут описаны каждая из отобранных моделей, их свойства и особенности обучения.

2.4.1 Модель I

Модель I имеет самую низкую ошибку в предсказаниях среди всех обученных моделей (6,6%). CHH состоит из 18 обучающихся слоёв. Всего в нейронной сети $\approx$ 2,7 млн. параметров.

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.



а) б)

Рисунок 3 — Соединение maxout активаций в модели I

Нейронная сеть обучалась 90 000 итераций методом стохастического градиентного спуска, 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.

2.4.2 Модели II и III

Модель II имеет в 4,5 раза меньше параметров, в сравнении с моделью I. Их число составляет $\approx 600\,000$. Данная нейронная сеть имеет классическую архитектуру, — чередование свёрточных и «пулинг» слоёв, в качестве функции активации используется Leaky ReLU[6] с параметром $\alpha = 0{,}01$.

Сеть обучалась в течении 30 000 итераций. Функционал ошибки оптимизировался алгоритмом Нестерова. Результат модели II оказался ниже на тестовом множестве, в сравнении с моделью I (89,8% против 93,4%). Данный результат объясняется недообучением, что является следствием недостаточного числа параметров нейронной сети. С другой стороны, за счёт уменьшения числа весов в модели, удалось снизить время обучения в 10 раз (с десяти часов до одного).

Модель III является попыткой исправить главный недостаток модели II за счёт увеличения числа параметров. В связи с чем, число фильтров каждого свёрточного слоя было увеличено в два раза, что позволило снизить ошибку на 1,3%.

2.4.3 Модели IV и V

На моделях IV и V проверялась гипотеза, основная идея которой в том, что pooling слои могут быть заменены на соответствующие свёрточные слои с увеличенным параметром stride без значительного снижения точности распознавания. [9]. В данных сетях max-pooling 2×2 слои были заменены на свёрточные слои со stride равным двум.

Модель IV имеет ≈ 6 млн. параметров и 13 слоёв. В качестве функции активации используется ReLU. После обучения в течении 50 000 итераций, точность на тестовом множестве достигла 92,1%, что подтверждает гипотезу о замене pooling слоёв.

2.4.4 Модель VI

Модель VI состоит из 18 слоёв и $\approx 4{,}2$ млн. параметров. Отличительной особенностью данной модели является слой batch normalization, который преобразовывает активаций свёрточных и полносвязных слоев таким образом, что их среднее равно нулю, а дисперсия единице (значения вычисляются на всём mini batch). Авторы batch normalization [10] утверждают, что данный приём приводит к общему

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

ускорению обучения и более быстрой сходимости, что действительно подтвердилось в данном эксперименте. Модель VI достигла точности в 91,8% за 25 0000 итераций, а общее время затраченное на обучение (40 000 итераций) составило всего 6 часов.

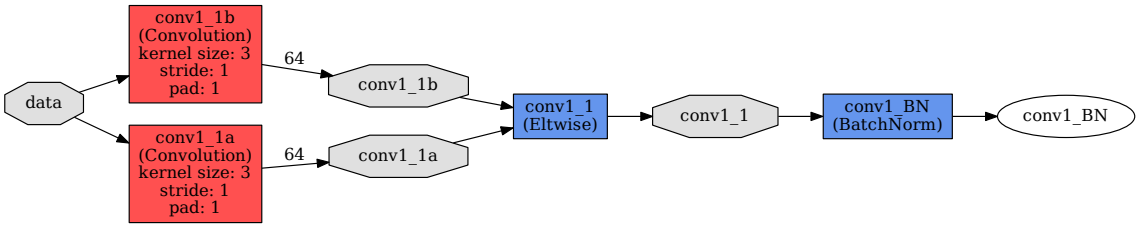


Рисунок 4 — Batch normalization слой в модели VI

2.4.5 Модель VII

Модель VII идентична модели I, за исключением количества свёрточных слоёв соединённых с maxout активацией. В данной модели их пять (Рис. 5), тогда как модель I имеет только два слоя. Такое изменение в топологии сети не привело к увеличению конечной точности модели, более того, ошибка увеличилась на 1,1%. Максимальное число итераций достигло 70 000, время затраченное на обучение составило 22 часа.

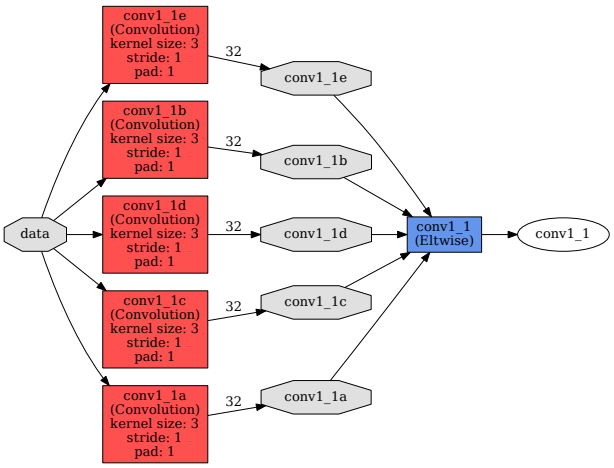


Рисунок 5 — Соединение maxout активаций в модели VII

2.5 Объединение нейронных сетей в ансамбль

Объединение нескольких моделей в ансамбль является мощным приёмом при решении различных задач машинного обучения. Применение подобной техники

было выбрано по причине наличия различных свёрточных нейронных сетей, предсказания которых не очень сильно коррелируют между собой. В данном исследовании модели объединялись с помощью «стэкинга» (stacking) [11], идея которого заключается в построении алгоритма, который делает финальные предсказания, основываясь на ответах других моделей (в данном случае на ответах нейронных сетей). Такой подход зачастую работает лучше любой из одиночных моделей.

Для построения конечного ансамбля были протестированы различные алгоритмы классификации, такие как случайный лес, метод k ближайших соседей, персептрон и SVM. Наилучшая точность была достигнута с помощью SVM (94,15%) с радиально-базисной функцией ядра (RBF kernel function). Конечная схема классификатора отражена на рисунке 6.

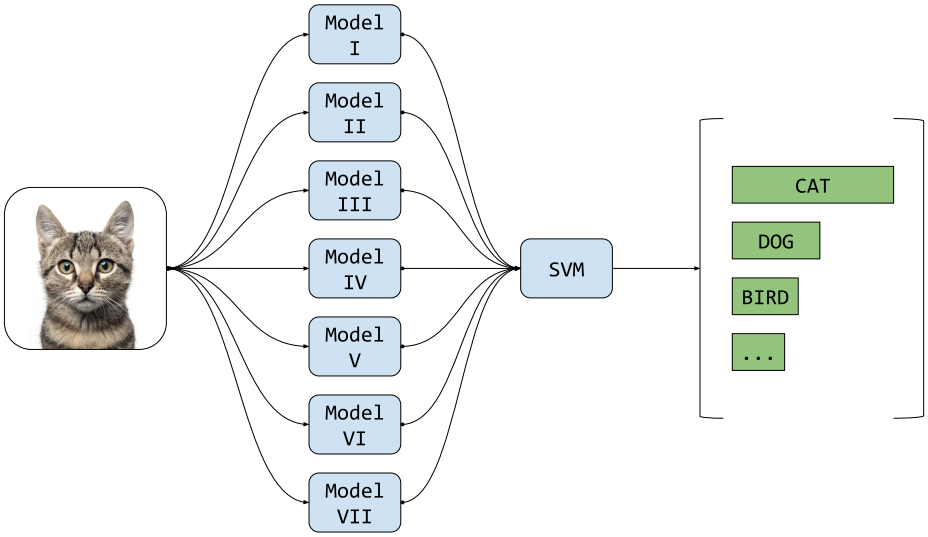


Рисунок 6 — Схема конечного ансамбля моделей

3 Заключение

Список литературы

1. *Sutskever Ilya*. A Brief Overview of Deep Learning. — 01.13.2015. http://goo.gl/lQVVmo.

2. *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.

3. 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.

4. *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.

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

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

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

8. 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.

9. Striving for Simplicity: The All Convolutional Net / Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, Martin A. Riedmiller // *CoRR*. — 2014. — Vol. abs/1412.6806. http://arxiv.org/abs/1412.6806.

10. *Ioffe Sergey, Szegedy Christian*. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift // *CoRR*. — 2015. — Vol. abs/1502.03167. http://arxiv.org/abs/1502.03167.

11. *Wolpert David H.* Stacked Generalization // *Neural Networks*. — 1992. — Vol. 5. — Pp. 241–259.

Приложение

Model I (93,4%)	Model II (89,8%)	Model III (91,1%)	Model IV (92,1%)	Model V (89,4%)	Model VI (91,8%)	Model VII (92,5%)
conv1_1 3x3x32	conv1_1 3x3x32	conv1_1 3x3x64	conv1_1 3x3x96	conv1 3x3x32	conv1_1 3x3x64	conv1_1 3x3x32
conv1_2 3x3x32	conv1_2 3x3x32	conv1_2 3x3x64	conv1_2 3x3x96	conv2 3x3x32	conv1_2 3x3x64	conv1_2 3x3x32
conv1_3 3x3x32	conv1_3 3x3x32	conv1_3 3x3x64	conv1_3 3x3x96	conv3 3x3x64	conv1_3 3x3x64	conv1_3 3x3x32
conv1_4 3x3x48	pool 2x2	pool 2x2	conv1_4 3x3x96 stride 2	conv4 3x3x64	conv1_4 3x3x64	conv1_4 3x3x48
conv1_5 3x3x48	conv2_1 3x3x64	conv2_1 3x3x128	conv2_1 3x3x192	conv5 3x3x128	conv1_5 3x3x64	conv1_5 3x3x48
pool 2x2	conv2_2 3x3x64	conv2_2 3x3x128	conv2_2 3x3x192	conv6 3x3x128	pool 2x2	pool 2x2
conv2_1 3x3x80	conv2_3 3x3x64	conv2_3 3x3x128	conv2_3 3x3x192	conv7 3x3x256	conv2_1 3x3x128	conv2_1 3x3x80
conv2_2 3x3x80	conv2_4 3x3x64	conv2_4 3x3x128	conv2_4 3x3x192 stride 2	conv8 3x3x256	conv2_2 3x3x128	conv2_2 3x3x80
conv2_3 3x3x80	pool 2x2	pool 2x2	conv3_1 3x3x384	conv9 3x3x256	conv2_3 3x3x128	conv2_3 3x3x80
conv2_4 3x3x80	conv3_1 3x3x128	conv3_1 3x3x256	conv3_2 3x3x384	conv10 3x3x256	conv2_4 3x3x128	conv2_4 3x3x80
conv2_5 3x3x80	conv3_2 3x3x128	conv3_2 3x3x256	conv3_3 3x3x384	conv11 3x3x256	conv2_5 3x3x128	conv2_5 3x3x80
conv2_6 3x3x80	conv3_3 3x3x128	conv3_3 3x3x256	conv3_4 3x3x384 stride 2	conv12 3x3x512	conv2_6 3x3x128	conv2_6 3x3x80
pool 2x2	conv3_4 3x3x128	conv3_4 3x3x256	pool global max	conv13 3x3x512	pool 2x2	pool 2x2
conv3_1 3x3x128	pool global ave	pool 2x2	fc 500	conv14 3x3x512	conv3_1 3x3x256	conv3_1 3x3x128
conv3_2 3x3x128	fc 256	fc 256	softmax 10	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 max					pool global ave	pool global ave
fc 500					fc 512	fc 512
softmax 10					softmax 10	softmax 10

Таблица 1: Архитектуры обученных моделей

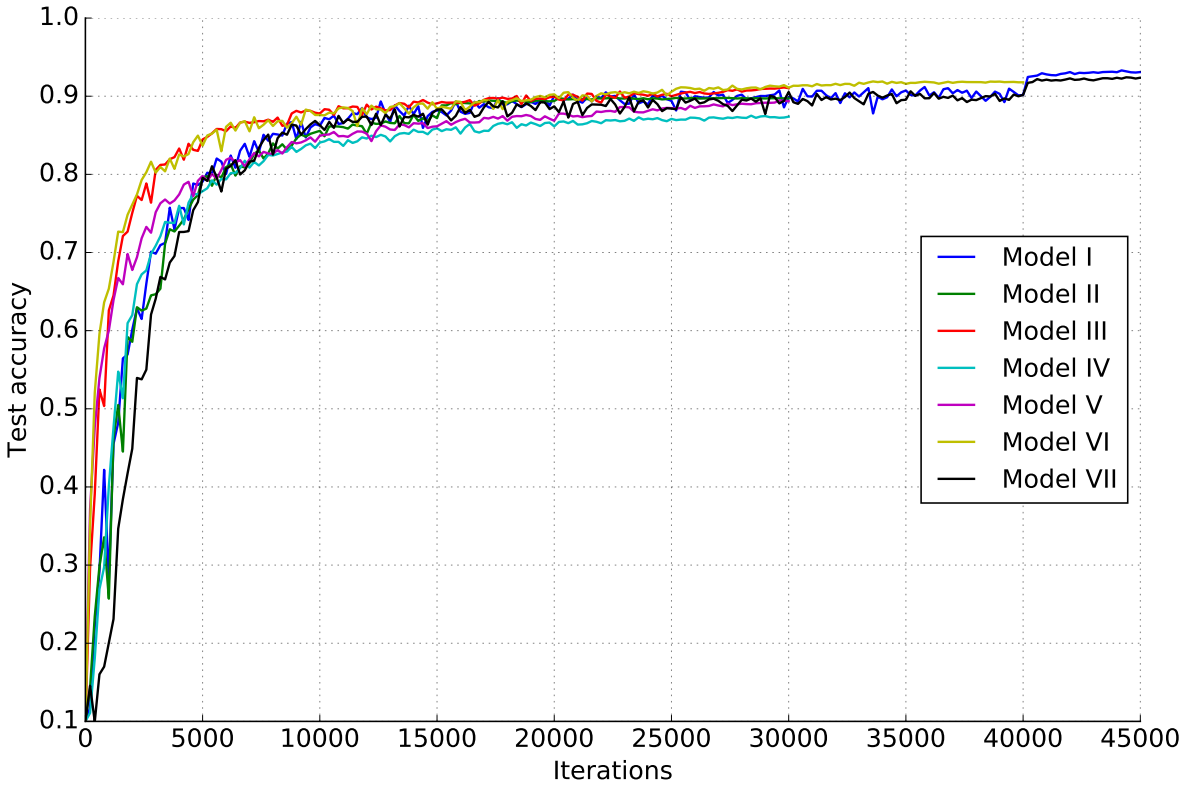


Рисунок 7 — Графики обучения