

Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
Волгоградский государственный технический университет

Факультет Электроники и вычислительной техники

Кафедра Системы автоматизированного проектирования и ПК

Согласовано

\_\_\_\_\_  
(должность гл. специалиста предприятия)

\_\_\_\_\_  
(подпись)      \_\_\_\_\_  
(инициалы, фамилия)  
«\_\_\_\_\_» \_\_\_\_\_ 2017

Утверждаю

Зав. кафедрой САПР и ПК, д.т.н.,

??.

\_\_\_\_\_  
(подпись)      М. В. Щербаков  
(инициалы, фамилия)  
«\_\_\_\_\_» \_\_\_\_\_ 2017

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к \_\_\_\_\_ выпускной работе бакалавра \_\_\_\_\_ на тему  
(наименование вида работы)

Портирование сверточной нейросети на ARM архитектуру с  
ограниченными вычислительными ресурсами и ресурсами памяти

Автор \_\_\_\_\_ Мельников Тимофей Алексеевич  
(подпись и дата подписания) (фамилия, имя, отчество)

Обозначение ВСТАВИТЬ КОД-81  
(код документа)

Группа ИВТ-461  
(шифр группы)

Направление ???.???.?? Автоматизированные системы управления  
(код по ОККО, наименование направления, программы)

Руководитель работы \_\_\_\_\_ А. В. Катаев  
(подпись и дата подписания) (инициалы и фамилия)

Консультанты по разделам:

_____ (краткое наименование раздела)	_____ (подпись и дата подписания)	_____ (инициалы и фамилия)
_____ (краткое наименование раздела)	_____ (подпись и дата подписания)	_____ (инициалы и фамилия)
_____ (краткое наименование раздела)	_____ (подпись и дата подписания)	_____ (инициалы и фамилия)

Нормоконтролер \_\_\_\_\_ ????? ?????????????  
(подпись и дата подписания) (инициалы и фамилия)

Волгоград, 2017

Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
Волгоградский государственный технический университет

Кафедра Системы автоматизированного проектирования и ПК

Утверждаю

Зав. кафедрой САПР и ПК, д.т.н.,

??.

\_\_\_\_\_  
(подпись) М. В. Щербаков  
(инициалы, фамилия)  
«\_\_\_\_\_» \_\_\_\_\_ 2017

Задание на \_\_\_\_\_ выпускную работу бакалавра

(наименование вида работы)

Студент \_\_\_\_\_ Мельников Тимофей Алексеевич

(фамилия, имя, отчество)

Код кафедры \_\_\_\_\_ ?? ?? Группа \_\_\_\_\_ ИВТ-461

Тема Портирование сверточной нейросети на ARM архитектуру с ограниченными вычислительными ресурсами и ресурсами памяти

Утверждена приказом по университету от «??» ?????? 201? № ????-ст

Срок представления готовой работы \_\_\_\_\_

(дата, подпись студента)

Исходные данные для выполнения работы

задание, выданное научным руководителем с кафедры САПР и ПК, утвержденное приказом ректора

Содержание основной части пояснительной записки

Что-то там раз

Что-то там два

Перечень графического материала

1) Графический материал раз

2) Графический материал два

ВСТАВИТЬ КОД-81

Руководитель работы \_\_\_\_\_

(подпись и дата подписания)

А. В. Катаев

(инициалы и фамилия)

Консультанты по разделам:

\_\_\_\_\_  
(краткое наименование раздела)

\_\_\_\_\_  
(подпись и дата подписания)

\_\_\_\_\_  
(инициалы и фамилия)

\_\_\_\_\_  
(краткое наименование раздела)

\_\_\_\_\_  
(подпись и дата подписания)

\_\_\_\_\_  
(инициалы и фамилия)

\_\_\_\_\_  
(краткое наименование раздела)

\_\_\_\_\_  
(подпись и дата подписания)

\_\_\_\_\_  
(инициалы и фамилия)

Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования

Волгоградский государственный технический университет  
Кафедра «Системы автоматизированного проектирования и ПК»

Утверждаю

Зав. кафедрой САПР и ПК, д.т.н.,

??.

_____	М. В. Щербаков
(подпись)	(инициалы, фамилия)
«_____»	_____ 2017

Портирование сверточной нейросети на ARM архитектуру с  
ограниченными вычислительными ресурсами и ресурсами памяти  
ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

ВСТАВИТЬ КОД-81

Листов 21

Научный руководитель  
старший преподаватель САПР и  
ПК

\_\_\_\_\_ А. В. Катаев  
«\_\_\_\_\_» \_\_\_\_\_ 2017

Нормоконтролер

?????, ????

\_\_\_\_\_ ????????????

«\_\_\_\_\_» \_\_\_\_\_ 2017

Исполнитель

студент группы ИВТ-461

\_\_\_\_\_ Т. А. Мельников

«\_\_\_\_\_» \_\_\_\_\_ 2017

Волгоград, 2017

Аннотация

Документ представляет собой пояснительную записку к выпускной работе бакалавра на тему «Портирование сверточной нейросети на ARM архитектуру с ограниченными вычислительными ресурсами и ресурсами памяти», выполненную студентом группы ИВТ-461, Мельниковым Тимофеем Алексеевичем.

В данной работе рассмотрена возможность реализации алгоритмов машинного обучения, в частности прямой проход сверточной нейронной сети, на устройстве с ограниченными вычислительными ресурсами и ресурсами памяти.

Объём пояснительной записки составил 21 страниц и включает 3 рисунков и 0 таблицы.

## Содержание

Введение	6
1 Обзор фреймворков машинного обучения	8
1.1 Caffe	8
1.1.1 Основные характеристики Caffe	9
1.1.2 Приемущества Caffe	10
1.1.3 Архитектура Caffe	10
1.2 Torch7	11
1.2.1 Основные характеристики Torch7	11
1.2.2 Структуры используемых данных	13
1.2.3 Пакеты Torch7	14
2 Используемые алгоритмы и модели	17
2.1 Теоретические основы нейронных сетей	17
2.1.1 Перцептрон - основа нейронных сетей	17
3 Проектирование системы	18
Заключение	19
Список использованных источников	20
Приложение А — Техническое задание	21

## Введение

Задачи обработки и анализа аналоговой информации являются одними из самых сложных в IT-индустрии. Долгое время такие задачи решались эвристическими линейными алгоритмами, которые требовали огромных аппаратных ресурсов при малой точности результата. На протяжении последних десяти лет стремительно растет и развивается прикладная область математики цель которой изучение и развитие искусственных нейронных сетей (НС). Актуальность разработок и исследований в данной области оправдывается применением НС в различных сферах деятельности. Это автоматизация процессов анализа объектов, образов, уневерсализация управления, прогнозирование, создание экспертных систем, анализ неформализованной информации и многие другие применения. В частности, в данной дипломной работе используются нейронные сети для классификации и детектирования объектов на изображении.

Наиболее существенным недостатком НС является их требовательность к вычислительным ресурсам и ресурсам памяти. Частично данная проблема решается использованием сверточных нейронных сетей, которые в виду особенностей логики работы позволяют в разы сократить потребляемые нейронной сетью ресурсы.

Не только искусственные нейронные сети являются трендом IT-индустрии, активно развивается концепция интернета вещей. Диапазон встраиваемых технологий простирается от концепции умных зданий до промышленной консолидации. Интеграция встраиваемых систем и искусственных нейронных сетей позволяет автоматизировать и упростить многие процессы во многих сферах деятельности.

В связи с вышесказанным целью данной дипломной работы является внедрение фрейворка машинного обучения на embedded систему C.H.I.P. и последующая его оптимизация. На основе проделанной работы необходимо сделать вывод о эффективности и рентабельности данного решения.

## ВСТАВИТЬ КОД-81

Для достижения поставленной цели необходимо решить следующие задачи:

- Изучить фреймворки глубокого машинного обучения
- Разработать консольное приложение для реализации прямого прохода нейронной сети
- Оптимизировать использование оперативной памяти и сделать загрузку весов по мере использования
- Разработать клиент-серверное приложение, демонстрирующее результат работы

В первом разделе пояснительной записки описаны фреймворки машинного обучения. Далее приведено обоснование выбора фреймворка darknet.

Во втором разделе описаны используемые модели нейронных сетей и алгоритм прямого прохода.

Третьей раздел посвящен разворачиванию фреймворка на устройстве С.Н.І.Р. и оптимизации работы алгоритма прямого прохода. Так же описана разработка клиент-серверной части для визуализации работы приложения.



## 1 Обзор фреймворков машинного обучения

Данный раздел содержит справочную информацию, технические особенности и функциональные возможности фреймворков глубокого машинного обучения и их сравнение. Так же раздел содержит обоснование выбора фреймворка darknet для встраивания и оптимизации на мобильном ПК С.Н.И.Р.

Из всего множества фреймворков были выделены Caffe, Torch7, Darknet, как наиболее зрелые, функционально полные и широко используемые.

### 1.1 Caffe

Caffe представляет собой фреймворк, разработанный учеными и практиками, с прозрачной и гибкой архитектурой для глубокого обучения и построения эталонных моделей. Фреймворк распространяется под BSD-лицензией и является с++ библиотекой. Так же реализованы обертки для python и MATLAB для универсализации обучения и развертывания глубоких моделей. Caffe используется на промышленных компаниях и в медиацентрах, обрабатывая 40 миллионов изображений в день на Titan GPU (примерно 2.5 миллисекунд на изображение). Одно из преимуществ Caffe это разделение модели данных от реализации. Что позволяет использовать приложения на разных платформах.

Caffe поддерживается и разрабатывается университетом Беркли, а именно центром BVLC.

### 1.1.1 Основные характеристики Caffe

Caffe представляет полный набор инструментов для обучения, тестирования, настройки и разработки моделей с подробной документацией и разобранными примерами. Поэтому процесс обучения использования фреймворка занимает короткий период. Возможность использования GPU делает Caffe одним из самых быстрых фреймворков, что позволяет его использовать в промышленном секторе. Такие показатели достигнуты благодаря особенностям описанным ниже.

Caffe является модульным программным обеспечением. Что позволяет легко добавлять новые форматы данных, слои и функции потерь. В фреймворке уже реализовано множество слоев и функций потерь, что позволяет реализовать нейронную сеть для задач различных предметных областей и категорий.

В Caffe представление и реализация разделены. Для описания модели в Caffe используется конфигурационный файл в формате `protobuf`. Caffe поддерживает сетевые архитектуры в форме произвольно ориентированных ациклических графов. Важным деталям является то, что после создания экземпляра модели Caffe выделяется ровно столько памяти, сколько необходимо для работы сериализованной нейронной сети и для хранения адреса объекта.[1]

В Caffe используется полное тестовое покрытие. Каждый модуль имеет собственный набор тестов. Модуль будет принят, только после прохождения всего набора тестов. Это позволяет эффективно оптимизировать модули и гарантирует стабильную работу фреймворка.

Caffe содержит предворительно обученные модели для академических целей и некоммерческого использования. Доступны сверточные НС с архитектурой "AlexNet" и вариации данной НС, обученные на базе данных ImageNet[2]. Так же доступны рекуррентные модели[3].

### 1.1.2 Приемущества Caffe

От других современных фреймворков глубокого обучения Caffe отличается следующими качествами(!):

- Реализация полностью основана на C++, что облегчает интеграцию с встраиваемыми системами. CPU режим позволяет использовать фреймворк без специализированного GPU.
- Готовые модели позволяют не тратить время и ресурсы на обучение. Важным пунктом является подробная документация для сериализации и использования моделей.

### 1.1.3 Архитектура Caffe

Caffe сохраняет и передает данные в четырехмерных массивах, которые названы блобами. Блобы представляют унифицированный интерфейс для работы памятью, содержащий пакеты изображений (или других данных), параметров или обновлений параметров. Блобы скрывают вычислительные издержки смешанной работы CPU и GPU, выполняя синхронизацию по мере необходимости. Память выделяется по требованию (лениво), что позволяет эффективней ее использовать. Модели сохраняются как буфер, использующий протокол Google (Google Protocol Buffers), который имеет ряд достоинств: минимальный размер строки при сериализации, эффективная сериализация, высокая читабельность в текстовом виде и удобные интерфейсы работы на нескольких языках. Необходимые для обучения огромные массивы данных хранятся в базах данных LevelDB. Google Protocol Buffers и LevelDB обеспечивают пропускную способность в 150 Мб/с.

Слой в Caffe представляет собой структуру соответствующую формальному определению слоя: он принимает на вход один или несколько блобов и выдает один или несколько блобов результатом. Caffe предоставляет полный набор типов слоев для глубокого обучения,

включая сверточный, pooling слой, inner products слой, нелиности, такие как выпрямленная линейная и логическая, слои потерь, таких как softmax и hinge. Настройка слоев требует минимальных усилий в виду композиционного построения сетей.

Caffe обеспечивает функциональность для любого направленного ациклического графа слоев, позволяя корректно выполнять прямой и обратный проход. Модели Caffe — это сквозные системы машинного обучения.[1]

## 1.2 Torch7

Torch7 — это универсальный математический фреймворк и библиотека машинного обучения, которая имеет оболочку для языка программирования Lua. Его цель — предоставить гибкую среду для проектирования и обучения моделей глубокого обучения. Гибкость достигается с помощью Lua, так как он является очень легким скриптовым языком. Эффективная реализация низкоуровневых числовых процедур, используя OpenMP и CUDA, позволяет фреймворку достиг высокой производительности. Фреймворк имеет простой Lua-интерфейс, что позволяет легко подключать его к стороннему программному обеспечению.

### 1.2.1 Основные характеристики Torch7

Структура фреймворка имеет три основных преимущества:

- она облегчает разработку численных методов;
- фреймворк легко расширяем (включая использование сторонних библиотек);
- высокая скорость работы фреймворка.

Второе преимущества достигается за счет выбранных разработчиками технологий. Скриптовый (интерпретируемый) язык с хорошим API-интерфейсом для C обеспечивает фреймворку гибкость в разработке и не накладывает ограничения на его расширяемость. Так как, язык высокого уровня делает процесс разработки программы более простым и понятным, чем язык низкого уровня. К тому же, интерпретируемость позволяет быстро и легко реализовывать различные идеи в интерактивном режиме. Хороший API-интерфейс сохраняет функциональные возможности из разных библиотек, так как становится прослойкой между универсальной структурой на языке Lua и различными структурами используемых библиотек на языке C.

Высокая скорость работы достигается благодаря компилятору JIT (Just In Time). На данный момент Lua является самым быстрым интерпретируемым языком. Lua разрабатывался для легкого внедрения в приложения, написанные на C. Поэтому представляет большое C-API на основе виртуального стека, для передачи значений между Lua и C. Это унифицирует интерфейс для C/C++ и делает обертывание библиотек тривиальным. [4]

Lua предназначен для использования в качестве мощного, легкого скриптового языка обладающими всеми необходимыми выразительными средствами. Он реализован как библиотека, которая написана на чистом C (точнее на подмножестве ANSI C и C++). Lua сочетает простой процедурный синтаксис с мощными конструкциями описания данных на основе ассоциативных массивов и расширяемой семантики. Lua динамически типизируется, выполняется путем интерпретации байт-кода для виртуальной машины на основе регистров и имеет автоматическое управление памятью с инкрементной сборкой мусора, что делает его идеальным для настройки, написания сценариев и быстрого прототипирования.[5]

Lua предлагает хорошую поддержку объектно-ориентированного программирования, функционального программирования и программирования, управляемого данными. Основным типом Lua является таблица, которая реализует ассоциативные массивы очень эффективным способом. Ассоциативный массив — это массив, который

может индексироваться не только числами, но и любыми другими типами данных языка. Таблицы не имеют фиксированного размера, они динамически изменяемы и могут использоваться как "виртуальные таблицы" над другой таблицей, что позволяет имитировать парадигмы объектно-ориентированного программирования. Таблицы являются единственным, но очень мощным механизмом структурирования данных в Lua. Torch7 использует таблицы для простого, равномерного и эффективного представления обычных массивов, таблиц символов, кортежей, очередей и других структур данных. Lua также использует таблицы для представления пакетов.

Lua и Python очень схожи как по структурированию данных, так и по стилю программирования. Если говорить о популярности в сообществе, то Python опережает Lua из-за огромного количества предоставляемых библиотек. Однако разработчики выбрали Lua по ряду других причин, которые, в виду специфики фреймворка, являются ключевыми. Во-первых, интеграция Lua с C очень проста. За несколько часов любая библиотека на C или C++ может стать библиотекой Lua. Во-вторых, Lua предоставляет эффективные возможности встраивания. Что бы преобразовать прототип в финальный продукт требуется не много дополнительной работы. В-третьих, Lua обладает высокой производительностью благодаря интерпритатору LuaJIT, который дает производительность на уровне C. Еще одним преимуществом Lua является переносимость. Lua написан на чистом ANSI C, его можно скомпилировать для любых устройств (сотовые телефоны, встроенные процессоры в FPGA, процессоры DSP и др.).

### 1.2.2 Структуры используемых данных

Ключевой сущностью в Torch7 является класс Tensor, предоставляемый автономной C-библиотекой Tensor. Данный класс расширяет базовый набор типов Lua, чтобы реализовать эффективную работу с многомерными массивами. Большинство пакетов Torch7 или

сторонних пакетов, зависящих от Torch7, реализуют собственный класс Tensor для представления сигналов, изображений, видео и других объектов, что делает интегрирование различных библиотек тривиальной задачей. Библиотека Torch Tensor предоставляет множество классических операций (включая операции линейной алгебры), которые реализованы и оптимизированы на C, используются SSE инструкции для Intel платформ. Дополнительно можно использовать высокопроизводительные реализации BLAS/Lapack операций линейной алгебры. Так же данная библиотека поддерживает инструкции OpenMP и вычисления на CUDA GPU.

### 1.2.3 Пакеты Torch7

На данный момент Torch7 имеет 7 основных пакетов:

- torch: основной пакет Torch7. Обеспечивает фреймворк классом Tensor, облегчает сереализацию и другие базовые функции;
- lap и plot: представляют стандартные функции для создания, преобразования и визуализации объектов Tensor. Пример работы показан на рисунке 1

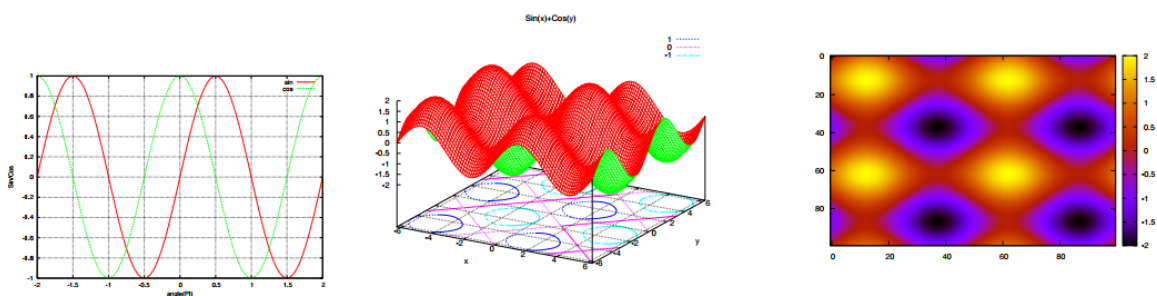


Рисунок 1 — Графики, полученные с помощью пакета plot фреймворка Torch7. Слева: простые синусоидальные функций. В центре: Поверхность, хранящаяся в 2D Tensor. Справа: Матричный график, построенный с использованием карты тепла

- qt: предоставляет интерфейс работы Torch7 с Qt. Реализует конвертацию Tensor в QImage и наоборот. Отлично подходит для

быстрого создания интерактивных демонстраций с кроссплатформенным графическим интерфейсом.

- `nn`: предоставляет набор стандартных модулей для создания нейронной сети. В пакет так же входит набор контейнерных модулей, которые можно использовать для определения произвольно направленных графов. Явное описание графа позволяет избежать сложности с анализатором графов или любого другого компилятора промежуточного уровня.

На практике нейронная сеть представляет собой последовательные графы, либо имеют шаблонные ветвления и рекурсии. На рисунке 2 показано создание многослойного перцептрона.

```
1 mlp = nn.Sequential()
2 mlp.add(nn.Linear(100,1000))
3 mlp.add(nn.Tanh())
4 mlp.add(nn.Linear(1000,10))
5 mlp.add(nn.SoftMax())
```

Рисунок 2 — Создание многослойного перцептрона, используя пакет `nn`

Каждый модуль или контейнер имеет стандартные функции для вычисления выходного состояния, обратного распространения производных входов и внутренних параметров. Для нейронной сети, приведенной на рисунке 2, вызов этих функций показан на рисунке 3.

```
1 Y = mlp.forward(X)           -- вычисление активации Y = f(X)
2 E = loss.forward(Y,T)        -- вычислить функцию потерь E = l(Y,T)
3 dE_dY = loss.updateGradInput(Y,T) -- вычислить градиент dE/dY = dl(Y,T)/dY
4 dE_dX = mlp.updateGradInput(X,dE_dY) -- вычислить ошибку, вплоть до dE/dx
5 mlp.accGradParameters(X,dE_dY) -- вычислить градиенты по весам: dE/dW
```

Рисунок 3 — Вычисление выходного состояния, обратного распространения производных входов и внутренних параметров

- `image`: пакет обработки изображений. Данный пакет предоставляет стандартные функции работы с изображениями (сохранение, загрузка, масштабирование, вращение, конвертация цветовых пространств, свертка и др.).

- `optim`: компактный пакет, который обеспечивает фреймворк методами оптимизации. В него входят реализация наклонного спуска,



сопряженного градиента и алгоритма Бroyдена — Флетчера — Гольдфарба — Шанно (BFGS).

- `unsup`: содержит алгоритмы обучения без учителя, такие как K-means, разреженное кодирование и автокодеры.

В дополнение к основным доступен постоянно растущий список сторонних пакетов. К примеру, `mattorch`, который обеспечивает двухсторонний интерфейс между матричным форматом Matlab и форматом Tensor или `parallel`, который предоставляет функции разветвления и исполнения Lua-кода на локальных или машинах, используя механизм сериализации Torch7. Этот список постоянно растет, поскольку Lua упрощает интерфейс любой библиотеки C.

ВСТАВИТЬ КОД-81

## 2 Используемые алгоритмы и модели

### 2.1 Теоретические основы нейронных сетей

#### 2.1.1 Перцептрон - основа нейронных сетей

В основе современной концепции

ВСТАВИТЬ КОД-81

### 3 Проектирование системы

ВСТАВИТЬ КОД-81

Заключение

Список использованных источников

- 1 <https://arxiv.org/pdf/1408.5093.pdf>
- 2 J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In ICML, 2014
- 3 A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, 2012
- 4 [http://ronan.collobert.com/pub/matos/2011\\_torch7\\_nipsw.pdf](http://ronan.collobert.com/pub/matos/2011_torch7_nipsw.pdf)
- 5 <http://www.lua.ru/doc/1.html>

ВСТАВИТЬ КОД-81

Приложение А  
Техническое задание