Санкт-Петербургский Государственный Университет

Математико-механический факультет

Кафедра системного программирования

Разработка и апробация инструментария iOSNeuron для решения задач распознавания изображений

Курсовая работа студента 445 группы

Манаева Дмитрия Сергеевича

Научный руководитель А.Е. Торегожин

Санкт-Петербург

2011

Оглавление

Введение

Искусственная нейронная сеть – это математическая модель (или вычислительная модель), основанная на структуре и функциональных аспектах биологических нейронных сетей. Искусственная нейронная сеть состоит из группы связанных между собой искусственных нейронов. В большинстве случаев это изменяющаяся система, которая изменяет свою структуру на основе информации, проходящей по нейронной сети во время её обучения. Как правило, они используются для моделирования сложных связей между входами и выходами или для нахождения путей.

В настоящее время искусственные нейронные сети применяются во многих областях. С их помощью решаются разнообразные задачи в сферах управления, прогнозирования, оценки качества.

Нейронные сети используются для распознавания образов. Они могут выделять определённые объекты из видеоряда, искать нужные объекты на фотографиях, распознавать лица. Нейронные сети часто используют в задачах идентификации личности.

Ещё одна задача для нейронных сетей – это выполнение прогнозов. В метеорологии на основе данных о температуре, влажности, скорости ветра, и т.д. с помощью нейронных сетей производятся прогнозы погоды (при этом, как правило, для разных сезонов используют разные нейронные сети). Использование нейронных сетей в финансовой сфере позволяет предсказывать динамику цен производственных финансовых активностей и индексов курсов акций, эффективность диверсификации портфельных капиталовложений, риски предоставления кредитов, а так же многое другое.

Таким образом, существует значительное количество приложений, использующих нейронные сети.

Последнее время всё большей популярностью пользуются мобильные устройства. Трудно представить себе человека, который не имел бы у себя смартфона, коммуникатора или хотя бы мобильного телефона. С развитием технологий мобильные устройства становятся всё более и более сложными, их функциональность разрастается, появляются новые приложения. Многие приложения используют нейронные сети, однако на данный момент инструментарии для создания и обучения искусственных нейронных сетей существуют преимущественно для компьютерных приложений. Таким образом, каждый разработчик мобильных приложений должен создавать и обучать нейронную сеть для своего приложения самостоятельно.

Инструментарий для создания нейронных сетей на мобильных устройствах сильно бы упростил работу разработчиков. Им надо было бы только задать параметры нейронной сети, предоставить обучающие данные, и всё остальное было бы выполнено за них.

Обзор существующих решений

На сегодняшний день наиболее мощными и перспективными инструментами, предоставляющими высокоуровневые средства для проектирования искусственных нейронных сетей, являются:

1) NeuroSolutions [1] – универсальный нейропакет фирмы NeuroDimension;

2) Neuroph [2] – проект с открытым исходным кодом, написанный на Java;

3) OpenCV [3] – проект с открытым исходным кодом, написанный на языке C++ и обеспечивающий основные функции нейронных сетей;

4) PWNLIB [4] – продукт отечественной компании «Павлин Технологии», обеспечивающий основные функции работы с нейронными сетями;

5) NNGPULIB [5] — продукт отечественной компании «Павлин Технологии», обеспечивающий ускорение расчета выхода многослойных нейронных сетей прямого распространения сигнала с применением графического процессора;

6) Neural Network Toolbox [6] – пакет дополнений к MATLAB, наиболее распространён и удобен для академических целей.

У перечисленных выше платформ существуют свои преимущества и недостатки. Neuroph является наиболее удобным в плане работы с нейронными сетями, так как включает в себя редактор визуального проектирования нейронных сетей, позволяющий проектировать нейронные сети намного быстрее. Преимуществом NeuroSolutions, NNGPULIB и OpenCV [7] является поддержка CUDA [8] (программно-аппартаная архитектура, позволяющая производить вычисления с использованием графических процессоров), что позволяет ускорять вычисления общего характера.

Основным недостатком всех вышеописанных инструментов является то, что они требуют слишком большого участия со стороны человека. От пользователя в большинстве случаев требуется выбрать тип нейронной сети, количество слоёв и нейронов [9], знание определённого языка программирования или специфики используемого инструмента.

Постановка задачи

Задачей данной курсовой работы является разработка инструментария для создания искусственных нейронных сетей на мобильных устройствах и его апробация для решения задач распознавания изображений.

Задача разделается на следующие подзадачи:

- добавление новых типов поддерживаемых нейронных сетей;
- разработка и реализация механизма сохранения и загрузки нейронных сетей;
- апробация механизма на примере задачи распознавания чисел.

Цель курсовой работы – получить некую библиотеку базовых классов, которую разработчик приложений сможет либо использовать для своих нужд такой, какая она есть, или же, пользуясь существующими классами, на их основе сконструировать что-то своё. Другими словами, он может либо воспользоваться уже реализованными типами нейронных сетей и алгоритмами обучения, либо написать свои, отнаследовав их от базовых классов.

Реализация

1. Технология

Прежде чем приступать к проектированию механизма, нужно выбрать платформу, на которой он будет реализован. Это связано с тем, что технологии ограничены в возможностях, и выбор той или иной технологии может накладывать ограничения на архитектуру самого инструмента. Для решения поставленной задачи технология должна удовлетворять следующим требованиям:

1) технология должна поддерживать C-подобные языки программирования. Данное требование обусловлено тем, что C-подобные языки являются самыми распространенными и быстрыми;

2) технология должна предоставить удобный способ работы с многопоточностью. Данное требоваие обусловлено тем, что многопоточность крайне важна для ИНС [10] ;

3) технология должна предоставлять возможность создавать инструменты для мобильных устройств. Данное требование обусловлено тем, что в последнее время мобильные устройства получили широкоераспространение,в связи с чем появилась потребность в программных ИНС-инструментах,работающих на мобильных устройствах [11].

Всем выше описанным требованиям удовлетворяет технология iOS SDK [12]:

1) технология поддерживает языки C, C++, Objective-C;

2) iOS SDK предоставляет удобный высокоуровневый механизм работы с многопоточностью [13];

3) программы, написанные при помощи iOS SDK, работают как на мобильных устройствах, так и на персональных компьютерах.

2. Архитектура iOSNeuron

Базовым классом для создания нейронной сети является класс NeuralNetwork, от которого должны наследоваться все типы нейронных сетей. В нем представлены все основные функции нейронных сетей. Наследники должны переопределять только инициализацию.

Нейронная сеть состоит из слоёв, каждый из которых, в свою очередь, состоит из нейронов. У каждого нейрона есть функция входа (InputFunction) и активационная функция (TransferFunction). На основе функции входа мы задаём изначальный вход нейрона.

Активационная функция используется при вычислении выхода нейрона и при обучении. InputFunction складывается на основе WeightsFunction и SummingFunction.

Для создания связей между нейронами используется класс Connection. Он содержит в себе вес, начальный и конечный нейроны. Сам нейрон хранит в себе следующие два множества:

1) множество InputConnections – все связи, которые входят в нейрон;

2) множество OutputConnections — все связи, которые выходят из нейрона;

У каждой нейронной сети можно задать обучающее правило — LearningRule. Данный класс также является базовым, от которого могут наследоваться другие классы, реализующие тот или иной алгоритм обучения. У каждого обучающего правила (LearningRule) есть обучающее множество (TrainingSet), которое, в свою очередь, состоит из обучающих элементов (TrainingElement). TrainingElement создаётся на основе обучающих данных (входного вектора и желаемого выхода).

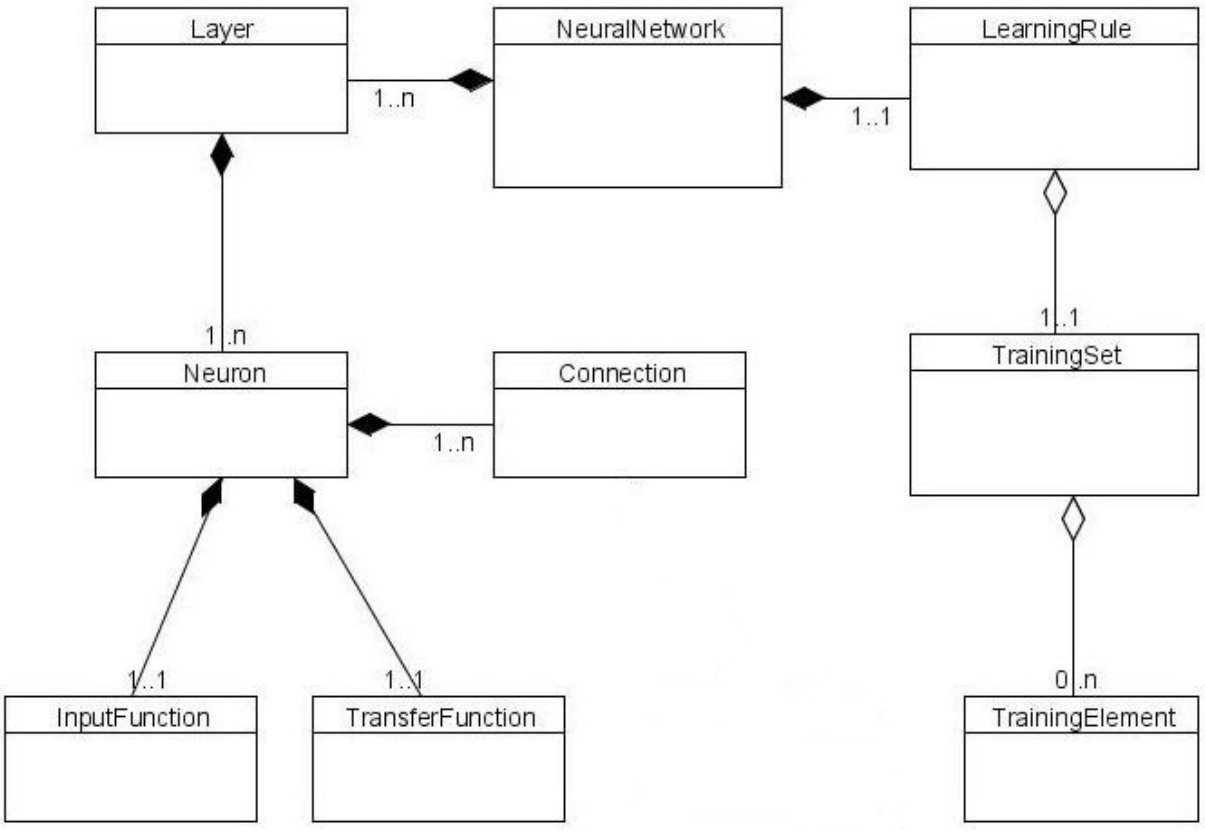

Рисунок 1:Архитектура iOSNeuron.

3. Сохранение и загрузка нейронных сетей

Так как процесс обучения нейронной сети является трудоёмким и занимает долгое время, имеет смысл не совершать его каждый раз, когда нам нужна сеть, а сделать один раз и запомнить результаты. Для этого было реализовано сохранение и загрузка нейронной сети в формате XML, что позволяет легко переносить и реализовывать архитектуру на других технологиях.

Для реализации сохранения был взят класс XMLTag, который позволяет генерировать xml файлы. Его реализация проста: создается тэг с определенным именем, к нему можно добавить атрибуты с помощью метода addAttributeWithName и внутренний текст через свойство (property) innerText. Внутренние тэги добавляются с помощью метода addChild. Сохраняется нейронную сеть при вызове метода save у базового класса NeuralNetwork. На текущий момент используется одна общая архитектура для хранения всех типов нейронных сетей.

Загрузка написана с использованием стандартного SAX-парсера NSXMLParser в классе NeuralNetworkXMLReader, который оказался очень удобным и быстрым. Работает как на iphone,так и на Mac OS.

```
<net>
<layer id = ”0” size = ”2”>
        <neuron id =”0” />
        <neuron id =”1” />
</layer>
<layer id = ”1” connectedTo = ”0” size = ”1”>
        <neuron id = ”0”>
                <connections>
                        <connection from = ”0”>0,5</connection>
                        <connection from = ”1”>-0,5</connection>
                </connections>
        </neuron>
</layer>
</net>
```

Выше представлен простой пример хранения нейронной сети. Корневым тэгом является тэг net — нейронная сеть. В нем хранится несколько слоев с именем layer, у

каждого слоя есть атрибуты id (идентификатор), size (количество нейронов) и connectedTo (указывает на идентификатор предыдущего слоя, с которым он связан). В layer лежат нейроны с уникальным идентификатором id для своего слоя. Связи же хранятся только у нейронов, в которых есть входные связи в форме тэга connection. Каждый connection имеет атрибут from, где указан уникальный идентификатор начального нейрона для этой связи из предыдущего слоя, внутри тэга лежит вес. Данная архитектура универсальна для хранения различных типов нейронных сетей.

4. Апробация iOSNeuron для решения задач распознавания картинок

4.1. Нейронные сети и решение задач распознавания картинок

Распознавание является одной из традиционных задач для нейронных сетей и они являются очень мощной технологией в этом направлении. В современном мире существуют следующие задачи, связанные с классификацией изображений:

- распознавание букв и чисел;
- распознавание штрих-кодов;
- расознавание автомобильных номеров;
- распознавание лиц и других биометрических данных (нейроидентификация личности).

Сферы применения задач по распознаванию образов очень обширны, и самый простой пример для их решения - это использование классических полносвязанных нейронных сетей прямого распространения, например, многослойного перцептрона с двумя или одним внутренним слоем.

На вход можно подавать матрицу исходного изображения в форме вектора и сделать один внутренний слой с большим количеством нейронов. Но так как размеры изображений в большинстве случаев не очень малы, а количество нейронов в внутреннем(-их) слое(-ях) должно быть довольно большим, соответственно, количество настраиваемых связей будет расти до сотен миллионов. Настраиваемые — это значит, что при обучении для каждой из них нужно будет вычислять градиент ошибки. Но это не всё, преобразуя изображение в цепочку байт, мы теряем топологию изображения, то есть взаимосвязь между отдельными его частями, причем с каждым слоем эта потеря усугубляется. Кроме того, нейронная сеть должна извлекать из данных некие инварианты, то есть задача распознавания подразумевает умение нейросети быть устойчивой к небольшим сдвигам, поворотам и изменению масштаба.

Решение этих проблем было найдено учеными Яном ЛеКуном [14] и Йошуа Бенгио, представившими концепцию свёрточных нейронных сетей по работам нобелевских лауреатов в области медицины Torsten Nils Wiesel и David H. Hubel [15]. Эти ученые проводили исследования зрительной коры головного мозга кошки [16].

4.2. Свёрточные нейронные сети

Идея сверточных нейронных сетей заключается в чередовании сверточных слоев (Convolution-layers), субдискретизирующих слоев (Subsampling-layers) и наличии полносвязных (Fullyconnected-layers) слоев на выходе.

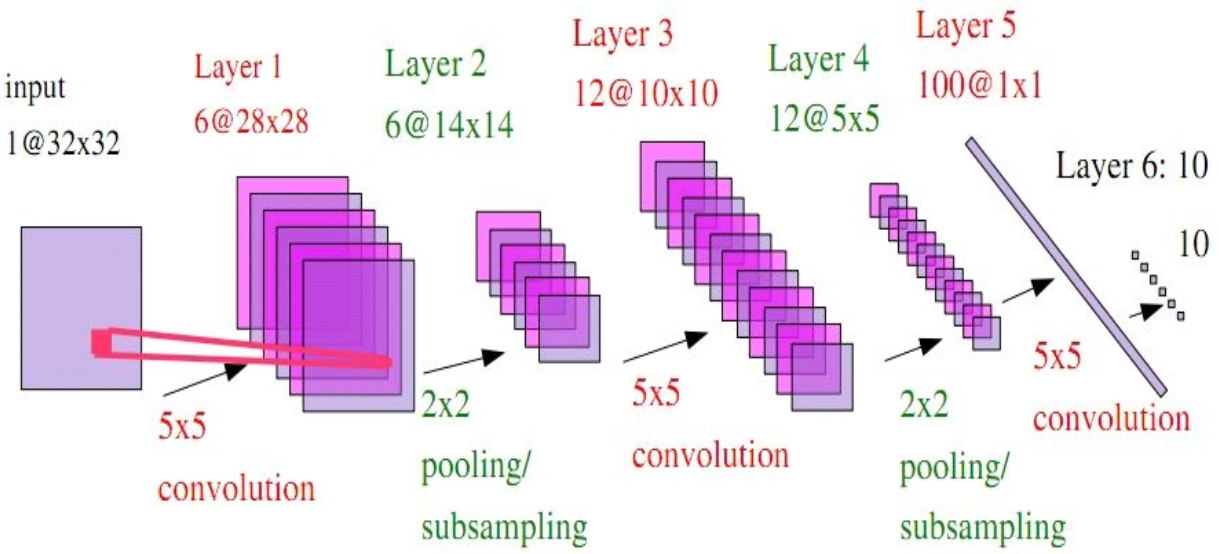

Рисунок 2: Пример 6-слойной сверточной нейронной сети.[18]

На рисунке 2 представлена сверточная сеть, где

Convolutional-layers – это слои, разбивающие исходное изображение (-ия) на карты признаков (feature map, уменьшенная и преобразованная картинка исходного), и каждый нейрон из одной карты признаков разделяет между собой одинаковые веса;

Pooling/subsampling-layers: - это слои, которые делают исходное изображение инвариантным к малым изменениям (уменьшению масштаба, сдвигам).

Сверточные нейронные сети заключают в себе 3 парадигмы:

1) Локальное восприятие;

2) Разделяемые веса;

3) Субдискретизация.

Особая структура сети позволяет сохранять топологию изображения от слоя к слою,

так как на вход одного нейрона (или выходы предыдущего слоя) подается не все изображение, а лишь некоторая его область. Такой подход называется локальным восприятием.

Концепция разделяемых весов предполагает, что для большого количества связей используется очень небольшой набор весов.

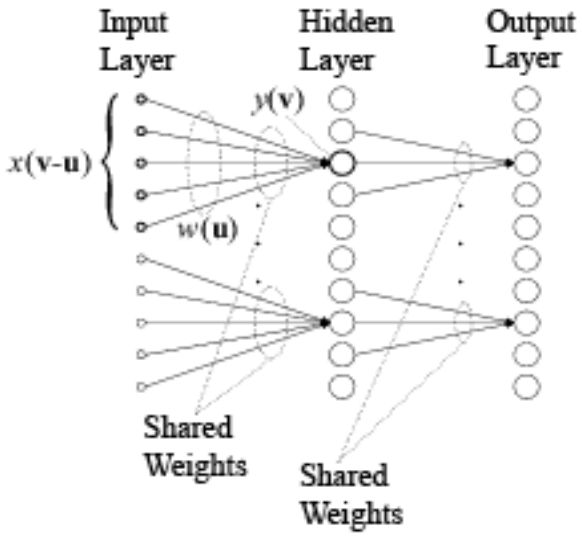

Рисунок 3: Разделяемые веса для одного слоя.[17]

На рисунке 3 представлен пример простой нейронной сети с разделяемыми весами, выделяющий одну инвариантную к изменениям (сдвигам) карту признаков.

Рассмотрим концепцию разделяемых весов на рисунке 2, у нас имеется на входе изображение размерами 32x32 пикселя, соответственно каждый из нейронов следующего слоя примет на вход только небольшой участок этого изображения размером 5x5, причем каждый из фрагментов будет обработан одним и тем же набором. Поскольку второй слой состоит из 6 карт признаков (изображений размером 28x28), здесь будет задействовано 6 наборов весов. Для сверточных нейронных сетей необходимо понимать, что самих наборов весов может быть много, но каждый из них будет применен ко всему изображению. Такие наборы часто называют ядрами (kernels). Таким образом, для 6 ядер размером 5x5 для входного изображения размерами 32x32 количество связей окажется равным примерно 120000 (что меньше 1 млн), а количество настраиваемых параметров всего 150. С другой стороны, ядро является фильтром при распознавании изображений, то есть некой матрицей коэффициентов. Эта матрица применяется к изображению с помощью математической операции, называемой свёрткой [19]. Суть этой операции заключается в том, что каждый фрагмент изображения умножается на матрицу (ядро) свертки поэлементно и результат

суммируется и записывается в аналогичную позицию выходного изображения. Основное свойство таких фильтров заключается в том, что значение их выхода тем больше, чем больше фрагмент изображения похож на сам фильтр. Таким образом, изображение, свернутое с неким ядром, даст нам другое изображение, каждый пиксел которого будет означать степень похожести фрагмента изображения на фильтр.

Ещё одна парадигма - субдискретизация, используемая на subsampling-layers, - состоит в том, что изображение грубо уменьшается в заданное число раз, что обеспечивает инвариантность к масштабу. Если вернуться к рисунку 2, можно заметить pooling/subsampling-layers, которые уменьшают изображение (-ия) предыдущего слоя ровно в 2 раза.

Чередуя сверточные и субдискритезирующие слоя, удается составлять карты признаков из карт признаков, что означает способность распознавания сложных иерархий признаков.

Благодаря вышеуказанным свойствам, сверточная нейронная сеть является лучшей в классификации изображений, если имеется небольшой набор тестовых данных для обучения. Особое применение она получила в визуальном анализе документов, идентификации личности, где её высокое качество подтверждается результатами, представленным в ряде научных работ:

1) “Character Recognition Using Convolutional Neural Networks”, David Bouchin 2007 [20];

2) “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”, Microsoft Research Patrice Y. Simard 2003 [22];

3) “Robust Face Analysis using Convolutional Neural Networks”, Beat Fasel 2001[23]

4) Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-Based Learning Applied to Document Recognition”, Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998. [46 pages] [21]

В результате для проверки работоспособности нашего инструментария (iOSNeuron) была выбрана и реализована задача распознавания чисел с использованием сверточных нейронных сетей. Для её реализации был создан класс для сверточных нейронных сетей ConvolutionalNN.

4.3. Реализация задачи распознавания чисел с использованием свёрточных нейронных сетей.

В процессе поиска решения данной задачи был найден источник [22], предоставляющий архитектуру, проверенную исследователями Microsoft Research. Для решения данной задачи существует база данных рукописных чисел MNIST [25], что значительно упрощает задачу. Она состоит из двух множеств:

- первое множество из 60,000 картинок рукописных чисел, используемых как обучающее множество для нейронной сети;

- второе множество из 10,000 картинок рукописных чисел, используемых для тестирования.

База данных состоит из 4 файлов: два файла idx3, где хранятся изображения рукописных чисел 28x28, и два файла idx1 с метками картинок. При работе с ней был написан класс MNIST, в котором производится побайтовое чтение и создание обучающего множества.

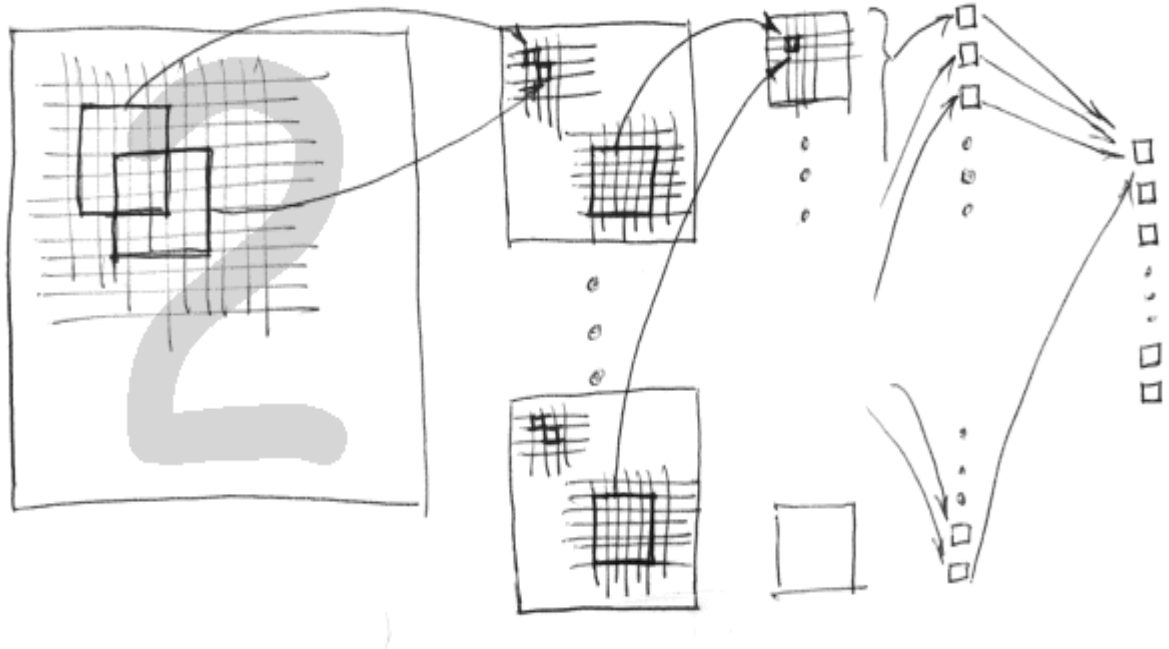

Рисунок 4: Архитектруа сверточной нейронной сети для распознавания рукописных чисел.

На рисунке 4 представлена архитектура, используемая при решении данной задачи. Первый слой это картинка 29x29, внутри которой центрировано число 20x20. Картинка

представлена в форме вектора из 841 нейрона. Следующий слой - это сверточный слой (convolutional-layer) с 6 картами признаков размером 13x13, где каждый нейрон соединен с исходным изображением ядром размером 5x5. В итоге во втором слое 13x13x6=1014 нейронов, (5x5+1)*6 = 156 весов, где + 1 это биас и 1014*26 =26364 связи.

Третий слой тоже сверточный и представлен в форме 50 карт признаков размером 5x5, где каждый нейрон соединен с каждой предыдущей картой признаков ядром размером 5x5. Таким образом, всего 50x5x5=1250 нейронов,50x6x(5x5+1)=7800 весов и 1250*26 = 32500 связей.

Четвертый и пятый слои являются полносвязанными. В четвертом 100 нейронов, (1250+1)x100= 125100 весов и связей. Пятый, последний слой состоит из 10 нейронов и 1010 параметров и связей.

Суммируя все данные об архитектуре, получим 3215 нейронов, 134066 весов и 184974 связи, что не очень много для нейронной сети.

Обучающее множество будет состоять из обучающих элементов, каждый из которых содержит в себе в форме входа одну картинку в оттенках серого, вектор размером 29x29, и на выходе нейронная сеть должна возвращать 1 в том нейроне, который является меткой к данной картинке, а остальные 9 нейронов вернут -1.

Результаты представленные в статье [22], гарантировали после обучения ошибку на тестовом множестве 0,82% <.

На основе этих данных была написана и обучена нейронная сеть для распознавания рукописных чисел.

Результаты работы

В данной работе достигнуты следующие результаты:

- дополнен инструментарий для разработка нейронных сетей на платформе iOS;
- реализована сверточная нейронная сеть;
- реализован механизм сохранения и загрузки сети;
- работоспособность инструментария проверена на примере решения задачи распознавания рукописных чисел.

Заключение

В рамках данной работы была осуществлена реализация и апробация инструментария для создания и обучения искусственных нейронных сетей для iOS. На данный момент количество разных типов создаваемых нейронных сетей мало, и в дальнейшем планируется расширение инструментария с помощью добавления других типов нейронных сетей. Так же возможно расширение алгоритмов обучения нейронных сетей.

Главный недостаток существующих инструментариев – слишком большое участие со стороны человека. Для того, чтобы воспользоваться тем или иным инструментом, необходимо знать его специфику или знать какой-то конкретный язык программирования, задавать количество слоёв в сети, количество нейронов в слое, выбирать тип сети и так далее. Планируется избавиться от этого недостатка и разработать алгоритм автоматизации построения нейронных сетей, который на основе обучающих данных будет сам выбирать тип сети, задавать её параметры, а на выходе выдавать уже готовую и обученную нейронную сеть.

Список литературы

1. NeuroSolutions http://www.neurosolutions.com/products/ns

2. Java Neural Network Framework http://neuroph.sourceforge.net/

3. OpenCV Wiki http://opencv.willowgarage.com/wiki/

4. ПАВЛИН ТЕХНОЛОГИИ – разработка систем с искусственным интеллектом, автоматизация производства, обработка данных и изображений – PWNLIB1.0 http://www.pawlin.ru/content/view/20/8/

5. ПАВЛИН ТЕХНОЛОГИИ - Программный модуль ускорения расчета выхода многослойных нейронных сетей прямого распространения сигнала с применением графического процессора — NNGPULIB 1.0 http://www.pawlin.ru/content/view/19/8/

6. Neural Network Toolbox - MATLAB http://www.mathworks.com/products/neuralnet/?BB=1

7. CUDA Wiki- http://ru.wikipedia.org/wiki/CUDA

8. NVIDIA ускоряет разработку OpenCV приложений с помощью GPU - http://www.nvidia.ru/object/nvidia-for-opencv-press-20100923-ru.html

9. Neural Network Toolbox - Documentation http://www.mathworks.com/help/toolbox/nnet/

10. А.И. Галушкин, Нейронные сети: основы теории

11. Kathryn Hymes, John Lewin, OCR for Mobile Phones, Stanford Feb/2009

12. iOS Dev Center – Apple Developer http://developer.apple.com/devcenter/ios/index.action

13. Concurrency Programming Guide: Introduction http://developer.apple.com/library/ios/#documentation/General/Conceptual/ConcurrencyProgrammingGuide/Introduction/Introduction.html

14. Yann LeCun's Home Page - http://yann.lecun.com/

15. Torsten Nils Wiesel – Wiki - http://en.wikipedia.org/wiki/Torsten_Wiesel

16. Haykin S. Neural Networks: A Comprehensive Foundation, second edition. Prentice Hall 1999. Chapter 4 Multilayer Perceptrons,pp. 156-255

17. Convolutional Neural Networks - Julio M. Otuyama http://www.inf.ufsc.br/~otuyama/eng/academic/cnn/index.html

18. Yann Lecun: “Convolutional Neural Networks presentation” 2010-07-12 http://www.cs.nyu.edu/~yann/talks/lecun-20100710-pcmi-03-convnets.pdf

19. Свёртка wiki- http://ru.wikipedia.org/wiki/Свёртка_(математический_анализ)

20. David Bouchin :“Character Recognition Using Convolutional Neural Networks”, 2007 - http://www.informatik.uni-ulm.de/ni/Lehre/WS06/SLTHS/ausarbeitungen/Bouchain.pdf

21. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-Based Learning Applied to Document Recognition”, Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998. [46 pages] - http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf

22. “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”, Microsoft Research Patrice Y. Simard, 2003 - http://research.microsoft.com/apps/pubs/?id=68920

23. “Robust Face Analysis using Convolutional Neural Networks”, Beat Fasel 2001 - http://publications.idiap.ch/index.php/publications/show/693

24. Y. LeCun, L. Bottou, G. Orr, and K. Muller, “Efficient BackProp” in Neural Networks: Tricks of the trade, (G. Orr and Muller K., eds.), 1998. - http://yann.lecun.com/exdb/publis/pdf/lecun-98b.pdf

25. MNIST handwritten digit database, Yann Lecun and Corina Cortes - http://yann.lecun.com/exdb/mnist/index.html