ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Институт компьютерных наук и технологий

**Высшая школа программной инженерии**

Изображение выглядит как текст

Автоматически созданное описание

**ПРАКТИЧЕСКАЯ РАБОТА №7**

по дисциплине «Проектирование интеллектуальных систем управления»

Студент А. М. Потапова

гр. 3530202/90202

Преподаватель Bahrami AmirHossein

Санкт-Петербург

2022 г

**Введение**

Необходимо рассмотреть пример сверточной нейронной сети. Данную нейронную сеть использовали для распознавания черно-белых рукописных чисел.

Изображение выглядит как текст

Автоматически созданное описание

**Ход работы**

1. Загрузка и изучение данных изображения

Загрузите цифровые образцы данных в качестве хранилища данных изображений.

Функция **imageDatastore** автоматически помечает изображения на основе имен папок и сохраняет данные в виде объекта **ImageDatastore**. Хранилище данных изображений позволяет хранить большие данные изображений, включая данные, которые не помещаются в памяти, и эффективно считывает пакеты изображений во время обучения сверточной нейронной сети.

Функция **fullfile** указывает полный путь к файлу с данными изображений. ‘**DigitDataset’** – название файла с данными изображений. Строки **‘matlabroot’**,**'toolbox'**,**'nnet'**,**'nndemos'**, **'nndatasets'** – директории.

digitDatasetPath = fullfile(matlabroot,'toolbox','nnet','nndemos', ... 'nndatasets','DigitDataset');

imds = 1(digitDatasetPath, ... 'IncludeSubfolders',true,'LabelSource','foldernames');

Подсчитайте количество изображений в каждой категории. Функция **countEachLabel** создает таблицу, содержащую метки и количество изображений, имеющих каждую метку. Хранилище данных содержит 1000 изображений для каждой из цифр 0-9, в общей сложности 10000 изображений.

labelCount = countEachLabel(imds)

|  |  |  |
| --- | --- | --- |
|  | Label | Count |
| 1 | 0 | 1000 |
| 2 | 1 | 1000 |
| 3 | 2 | 1000 |
| 4 | 3 | 1000 |
| 5 | 4 | 1000 |
| 6 | 5 | 1000 |
| 7 | 6 | 1000 |
| 8 | 7 | 1000 |
| 9 | 8 | 1000 |
| 10 | 9 | 1000 |

Далее мы указываем размер изображений для входной слой нейронной сети. Каждое изображение размером 28-28-1 пикселей.

img = readimage(imds,1);

1. Создание дата сетов для обучения и валидации нейронной сети.

Данные изображений разделяем на два дата сета по каждой категории. Обучающий дата сет содержит 750 изображений. Дата сет для проверки содержит все оставшиеся изображения. Функция splitEachLabel разделяет дата сет digitData на два новых дата сета trainDigitData и valDigitData.

ё

numTrainFiles = 750;  
[imdsTrain,imdsValidation] = splitEachLabel(imds,numTrainFiles,'randomize');

с

1. Настройка архитектуры сети.

Архитектура сверточной нейронной сети определена следующим образом:

layers = [ imageInputLayer([28 28 1])

convolution2dLayer(3,8,'Padding','same') batchNormalizationLayer  
reluLayer

maxPooling2dLayer(2,'Stride',2)

convolution2dLayer(3,16,'Padding','same') batchNormalizationLayer  
reluLayer

maxPooling2dLayer(2,'Stride',2)

convolution2dLayer(3,32,'Padding','same')

batchNormalizationLayer

reluLayer

fullyConnectedLayer(10)

softmaxLayer

classificationLayer

];

Описание каждого слоя:

* **Image Input Layer.**

Входной слой изображения — это место, где вы указываете размер изображения, который в данном случае составляет 28 на 28 на 1. Эти числа соответствуют высоте, ширине и размеру канала. Цифровые данные состоят из изображений в оттенках серого, поэтому размер канала (цветовой канал) равен 1. Для цветного изображения размер канала равен 3, что соответствует значениям RGB. Вам не нужно перетасовывать данные, потому что trainNetwork по умолчанию перетасовывает данные в начале обучения. trainNetwork также может автоматически перетасовывать данные в начале каждой эпохи во время обучения.

* **Convolutional Layer.**

В сверточном слое первым аргументом является размер фильтра, который представляет собой высоту и ширину фильтров, используемых обучающей функцией при сканировании изображений. В этом примере число 3 указывает на то, что размер фильтра составляет 3 на 3. Вы можете указать различные размеры для высоты и ширины фильтра. Второй аргумент — это количество фильтров, numFilters, которое представляет собой количество нейронов, подключающихся к одной и той же области ввода. Этот параметр определяет количество карт объектов. Используйте пару "Заполнение" имя-значение, чтобы добавить заполнение к карте входных объектов. Для сверточного слоя с шагом по умолчанию, равным 1, "одинаковое" заполнение гарантирует, что размер пространственного вывода совпадает с размером ввода. Вы также можете определить шаг и скорость обучения для этого слоя, используя аргументы пары имя-значение convolution2dLayer.

* **Batch Normalization Layer.**

Слои пакетной нормализации нормализуют активации и градиенты, распространяющиеся по сети, что упрощает задачу оптимизации сетевого обучения. Используйте слои пакетной нормализации между сверточными слоями и нелинейностями, такими как слои ReLU, чтобы ускорить обучение сети и снизить чувствительность к инициализации сети. Используйте слой пакетной нормализации для создания слоя пакетной нормализации.

* **ReLU Layer.**

За уровнем пакетной нормализации следует нелинейная функция активации. Наиболее распространенной функцией активации является выпрямленный линейный блок (ReLU). Используйте realplayer для создания слоя ReLU.

* **Max Pooling Layer.**

За сверточными слоями (с функциями активации) иногда следует операция понижающей дискретизации, которая уменьшает пространственный размер карты объектов и удаляет избыточную пространственную информацию. Понижающая выборка позволяет увеличить количество фильтров в более глубоких сверточных слоях без увеличения требуемого объема вычислений на слой. Одним из способов уменьшения выборки является использование максимального пула, который вы создаете с помощью maxPooling2dLayer. Уровень максимального объединения возвращает максимальные значения прямоугольных областей входных данных, указанные первым аргументом poolSize. В этом примере размер прямоугольной области равен [2,2]. Аргумент пары имя- значение 'Stride' задает размер шага, который принимает обучающая функция при сканировании входных данных.

* **Fully Connected Layer.**

За сверточными слоями и слоями с понижающей выборкой следуют один или несколько полностью связанных слоев. Как следует из названия, полностью связанный слой — это слой, в котором нейроны соединяются со всеми нейронами предыдущего слоя. Этот слой объединяет все особенности, изученные предыдущими слоями по всему изображению, чтобы идентифицировать более крупные узоры. Последний полностью связанный слой объединяет функции для классификации изображений. Следовательно, параметр выходного размера в последнем полностью подключенном слое равен количеству классов в целевых данных. В этом примере размер вывода равен 10, что соответствует 10 классам. Используйте fullyConnectedLayer для создания полностью связанного слоя.

* **Softmax Layer.**

Функция активации softmax нормализует выходные данные полностью подключенного слоя. Выходные данные слоя softmax состоят из положительных чисел, сумма которых равна единице, которые затем могут быть использованы классификационным слоем в качестве вероятностей классификации. Создайте слой softmax с помощью функции softmax Layer после последнего полностью подключенного слоя.

* **Classification Layer.**

Последний слой — это слой классификации. Этот уровень использует вероятности, возвращаемые функцией активации softmax для каждого входа, чтобы назначить вход одному из взаимоисключающих классов и вычислить потери. Чтобы создать слой классификации, используйте classificationLayer.

1. Установка параметров обучения сети.

После определения структуры сети мы указали параметры обучения. Обучили сеть, используя стохастический градиентный спуск с импульсом (SGDM) с начальной скоростью обучения 0,01. Установили максимальное количество эпох равным 4. Эпоха - это полный цикл обучения для всего обучающего набора данных. Контролируйте точность сети во время обучения, задавая данные проверки и частоту проверки. Перемешивали данные каждую эпоху. Программное обеспечение обучает сеть на основе обучающих данных и вычисляет точность данных проверки через регулярные промежутки времени во время обучения. Данные проверки не используются для обновления весовых коэффициентов сети. Включили график хода обучения и отключите вывод командного окна.

options = trainingOptions('sgdm', ...

'InitialLearnRate',0.01, ...

'MaxEpochs',4, ...

'Shuffle','every-epoch', ...

'ValidationData',imdsValidation, ...

'ValidationFrequency',30, ...

'Verbose',false, ...

'Plots','training-progress');

1. Обучение нейронной сети с помощью обучающего дата сета.

График прогресса обучения показывает потери и точность мини-партии, а также потери и точность проверки. Дополнительные сведения о графике прогресса обучения см. в разделе Мониторинг прогресса обучения глубокому обучению. Потеря — это кросс-энтропийная потеря. Точность — это процент изображений, которые сеть классифицирует правильно.

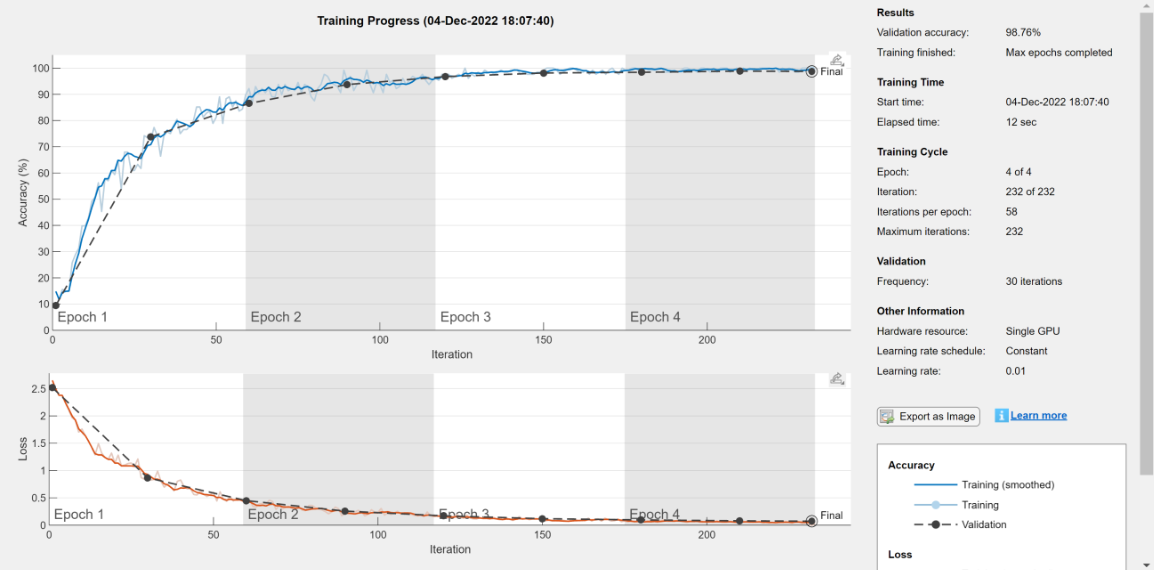
net = trainNetwork(imdsTrain,layers,options);

1. Распознавание изображений на дата сете для проверки и вычисление точности.

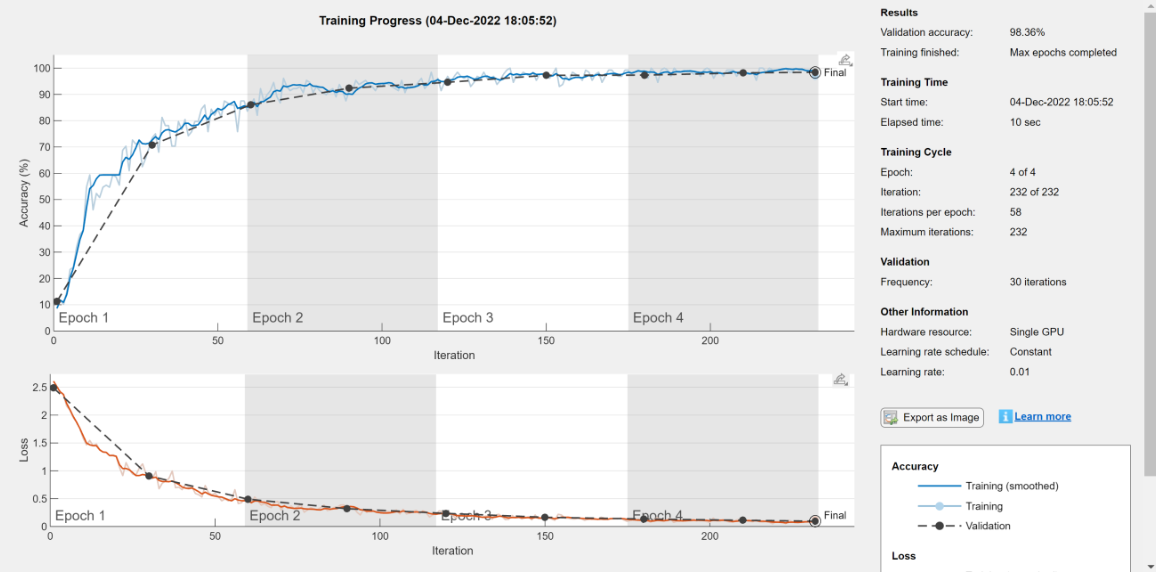
YPred = classify(net,imdsValidation);

YValidation = imdsValidation.Labels;

accuracy = sum(YPred == YValidation)/numel(YValidation)



Попробуем изменить параметры нейронной сети:  
Для начала попробуем понизить количество фильтров для первого сверточного слоя:



Точность немного уменьшилась.  
Попробуем добавить еще один сверточный слой:

layers = [  
imageInputLayer([28 28 1])

convolution2dLayer(3,3,'Padding','same')

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,'Stride',2)

convolution2dLayer(3,16,'Padding','same')

batchNormalizationLayer

reluLayer

maxPooling2dLayer(2,'Stride',2)

convolution2dLayer(3,32,'Padding','same')

batchNormalizationLayer  
reluLayer  
maxPooling2dLayer(2,'Stride',2)

convolution2dLayer(3,64,'Padding','same')

batchNormalizationLayer

reluLayer

fullyConnectedLayer(10)

softmaxLayer classificationLayer];

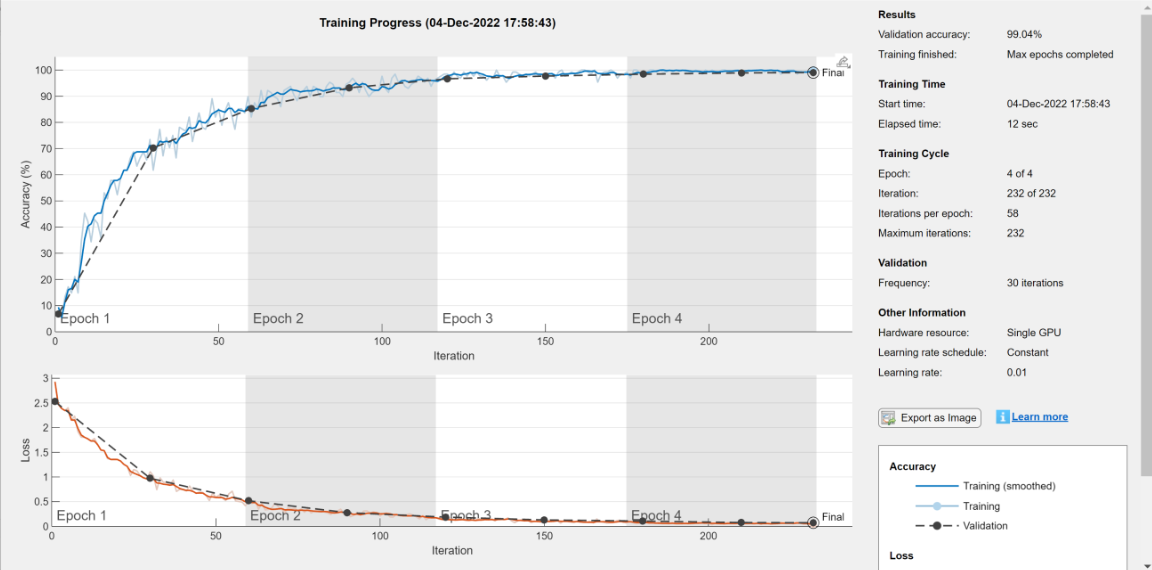
Добавили еще 4 дополнительных слоя сверточной сети:

maxPooling2dLayer(2,'Stride',2)

convolution2dLayer(3,64,'Padding','same')

batchNormalizationLayer  
reluLayer

За счет добавления еще одного этапа свертки с большим количеством фильтров точность немного увеличилась.



**Вывод**

В ходе выполнения этого задания мы познакомились с интерфейсом и возможностями neural network toolbox, разобрали один из примеров обучили нейронную сеть и провели классификацию. Так же изменили некоторые параметры нейронной сети и рассмотрели полученные результаты.