Лабораторная работа 4

Применение сверточных нейронных сетей для классификации изображений

Цель работы

Получение теоретических знаний и практических навыков в проектировании, настройке и использовании сверточных нейронных сетей (convolutional neural networks) для классификации изображений на основе существующих библиотек и фреймворков – Keras, Tensorflow (Python), Deep Learning Toolbox (Matlab).

Описание работы

Глубокое обучение

В 2000-2010х произошел стремительный технологический скачок в области нейронных сетей, связанный с рядом факторов:

- разработка достаточно эффективных с точки зрения обучения архитектур нейронных сетей для решения задач распознавания образов (изначально это в основном касалось классификации изображений);
- создание универсальных программных библиотек для работы с нейронными сетями (создание, обучение, вспомогательные функции) – это позволило множеству исследователей достаточно просто создавать и обучать нейронные сети для конкретной задачи, не тратя при этом большую часть времени на программирование и математические выкладки;
- применение мощностей видеопроцессоров (CUDA) для распараллеливания процесса обучения и функционирования нейронных сетей, что позволило обучать весьма большие по количеству слоев, нейронов и связей нейронные сети, что до этого было невозможно.

Если раньше нейронные сети использовали 1, максимум 2 скрытых слоя, то сейчас их количество может даже превосходить сто и более. В связи с этим такие нейронные сети стали называть глубокими, а сам процесс создания и обучения таких сетей – глубоким обучением.

Процесс синтеза, настройки и обучения нейроннных сетей превратился в технологию, немного вытеснив науку.

Библиотеки и фреймворки для глубокого обучения

Количество различных библиотек и фреймворков для машинного и глубокого обучения постоянно изменяется. В таблице 1 приведены наиболее распространенные и известные фреймворки. Часть из библиотек на текущий момент поглощена другими, в частности Keras является частью Tensorflow, а Caffe2 – частью Pytorch.

Таблица 1

Сравнение фреймворков и библиотек глубокого обучения

	Разработчик	Открытое	Язык	Развивается
tensorflow	Google	+	C++, Python	+
keras	Ф. Шолле	+	Python	+
caffe	Berkeley univercity	+	C++	-
theano (lasagne)	Monreal univercity	+	Python	-
deeplearning4j	San-Francisco	+	Java	
torch	Ронан Коллобер, Корай Кавукчуоглу, Клеман Фарабе	+	C	-
		+		
PyTorch (Caffe2)	Facebook	+	Python, C++, Cuda	+
Flux	Mike Innes	+	Julia	+
MXnet	Apache	+	C++	+
Microsoft Cognitive Toolkit (CNTK)	Microsoft	+	C++	-
Matlab+Deep Learning Toolbox	Mathworks	-	C, C++, Java, Matlab	+
Wolfram Mathematica	Mathematica	-	C++, Wolfram, CUDA	+
OpenNN	Artelnics	+	C++	+
PlaidML	Vertex AI, Intel	+	Python, C++, OpenML	+

Наличие большого количества фреймворков приводит к проблеме совместимости, в частности моделями, сделанными в одном из фреймворков, зачастую затруднительно пользоваться в другом. Поэтому был разработан открытый стандарт по обмену нейронными сетями (Open Neural Network Exchange, ONNX), позволяющий конвертировать модели Caffe2, PyTorch, CNTK в унифицированный формат и использовать его затем в специальных т.н. компиляторах экосистем глубокого обучения. Одним из таких решений является библиотека nGraph от Intel. Она напрямую работает с моделями Tensorflow и MxNet и косвенно с рядом моделей через ONNX.

Существуют также специализированные фреймворки для работы с моделями глубокого обучения на мобильных устройствах – Tensorflow Lite (Google), Core ML (Apple), Firebase ML Toolkit, Clarifai, Fritz Ai, Skafos, Numericcal.

Модели глубокого обучения

Развитие современных фреймворков для создания и обучения различных моделей нейронных сетей привело к появлению большого количества моделей, эффективно решающих задачи распознавания образов – классификация, сегментация, детектирование объектов на изображениях. Большинство из этих моделей были обучены на массивных

базах данных (ImageNet) с применением суперкомпьютеров, что делает их достаточно универсальными в смысле всевозможных аспектов применения.

Многие из моделей доступны в краткой и детальной или предобученной форме. В первом случае это некоторая функция или класс, выполняющая создание на соответствующем фреймворке нужной модели. Во втором случае для модели дополнительно предоставляется файл с весами, в котором содержится информация обо всех весовых коэффициентах/смещениях/вспомогательных параметрах, которые были получены в результате обучения модели на некотором исходном наборе данных.

В таблице ниже приведены популярные и известные модели глубоких сверточных НС, применяемых для распознавания изображений.

Таблица 2

Популярные модели глубокого обучения для распознавания изображений

НС	Число слоев
alexnet	8
vgg16, vgg19	16, 19
resnet18, resnet50, resnet50v2, resnet101, resnet101v2, resnet152, resnet152v2	18, 50, 101, 152
inceptionv3	50
inceptionresnetv2	164
Xception	71
googlenet	22
squeezenet, SqueezeNext	18
densenet121, densenet169, densenet201	121, 169, 201
efficentnetB0-…-efficientnetB7	
mobilenet, mobilenetv2, mobilenetv3Large, mobilenetv3Small	28, 52
NASNetLarge, NASNetMobile	
MnasNet	
BlazeFace	
TinyYOLO / Darknet	
ShuffleNet	
CondenseNet	
ESPNet	
DiCENet	
FBNet & ChamNet	
GhostNet	

Дообучение

В случае, если модель доступна в детальной форме, то доступна возможность ее тонкой настройки (fine tuning), также носящей название "дообучение" или перенос обучения (transfer learning). В этом случае обучается (дообучается) только последний слой (или несколько последних слоев), а все начальные слои используются только для вычисления признаков, которые будут являтся входным сигналом для обучаемых слоев. Данный подход значительно сокращает временные и вычислительные ресурсы с одной

стороны и позволяет пользоваться возможностями предобученной модели по выделению самых различных признаков из изображений.

Дополнительные материалы, ссылки, источники

Зоопарк моделей Caffe
https://github.com/BVLC/caffe/wiki/Model-Zoo
Модели Tensorflow
https://github.com/tensorflow/models
Keras
https://keras.io/
Matlab Deep Learning Toolbox
https://www.mathworks.com/help/deeplearning/index.html?s_tid=CRUX_lftnav
Сравнение современных моделей
https://machinethink.net/blog/mobile-architectures/
Рейтинг моделей на основе ImageNet
https://paperswithcode.com/sota/image-classification-on-imagenet
База данных IMageNet
http://www.image-net.org/

Исходные данные

К п. 1 и 3 в качестве исходных данных используются многомерные образы – символы типов 1, 2 и 3(4) для своего варианта. В соответствии с файлом "Описание многомерных образов (символов) и искажений.xlsx" также необходимо рассмотреть различные варианты искажений этих символов.

В п 2 в качестве исходных данных используются данные, на которых выбранные модели обучались их авторами.

Инструментарий

В качестве фреймворка для глубокого обучения может быть выбран любой из указанных в таблице 1 при условии, что в них имеются выбранные предобученным модели глубоких НС.

Следует отметтть, что больше всего примеров и ресурсов имеется по связке – фреймворк Tensorflow и библиотека Keras. Также хорошую документацию имеет Matlab Deep Learning Toolbox и PyTorch (Caffe2).

Задание

1. Реализация простой глубокой НС.

1.1 Вначале необходимо реализовать простую модель глубокой сверточной НС с относительно небольшим суммарным количеством скрытых слоев (около 10) так, чтобы обучение можно было выполнить на среднем по производительности ПК.

НС должна состоять из нескольких последовательно соединенных слоев следующих типов

- сверточные слои Conv2D
- слои для улучшения качества обучения (регуляризация, прореживание, избежание переобучения) – pooling (MaxPool2d), dropout, dropconnect.
- слои с нормализацией (BatchNormalization)
- слои с преобразованием размерности (Flatten)
- слои с нелинейными передаточными функциями (RELU)
- полносвязные слои (dense)

Можно руководствоваться имеющимися примерами в книгах, статьях, мастер-классами и др, например

https://www.tensorflow.org/datasets/keras_example
https://keras.io/examples/vision/mnist_convnet/
https://www.kaggle.com/adityaecdrid/mnist-with-keras-for-beginners-99457

1.2. С помощью Keras/Tensorflow/Matlab создать программную модель НС

1.3. Сформировать обучающую и тестовую выборку для модели.

1.4. Выполнить обучение.

1.5. Проанализировать качество распознавания с помощью матрицы неточностей, привести примеры плохо различимых образов.

2. Изучение существующих моделей глубокого обучения.

2.1. Выбор моделей, загрузка весов.

Из таблицы 2 выше необходимо выбрать три различные модели глубоких нейронных сетей, отличающихся по сложности архитектуры – простая, средняя и сложная. Необходимо также, чтобы эти модели были предобучены на какой-либо базе данных (Imagenet, Cifar10) и веса можно было загрузить в открытом доступе.

2.2. Загрузка части данных изображений, на которых модель была обучена.

2.3. Создание моделей в Keras, загрузка весов.

2.4. Применение моделей для классификации изображений.

2.5. Проанализировать качество распознавания с помощью матрицы неточностей, привести примеры плохо различимых образов. Сравнить модели по эффективности, сложности и времени работы.

3. Дообучение (перенос обучения).

Воспользоваться тремя моделями из п. 2 и данными выборок из п.1

3.1. Загрузить модели и веса и попробовать выполнить классификацию. Проанализировать результаты.

3.2. Загрузить модели и веса таким образом, чтобы последний слой был обучаемым, а все предыдущие – нет.

3.3. Выполнить обучение последнего слоя на основе имеющихся выборок.

3.4. Применить модели для классификации изображений имеющихся тестовых выборок.

3.5. Проанализировать качество распознавания с помощью матрицы неточностей, привести примеры плохо различимых образов. Сравнить результаты с п. 1.5, 2.5 и 3.3.

4. Выводы и анализ результатов

4.1. Проанализировать результаты всей работы с помощью диаграмм, графиков, таблиц.

4.2 Написать выводы по сравнению различных подходов к классификации изображений с помощью глубоких сверточных НС.