УДК 004.89

ВЫБОР МЕТОДОВ ГЛУБОКОГО ОБУЧЕНИЯ ДЛЯ РЕШЕНИЯ ЗАДАЧИ РАСПОЗНАВАНИЯ БОЛЕЗНЕЙ РАСТЕНИЙ В УСЛОВИЯХ МАЛОЙ ОБУЧАЮЩЕЙ ВЫБОРКИ

Сметанин Артем Алексеевич[1], Гончаров Павел Владимирович[2], Ососков Геннадий Алексеевич[3]

[1]Студент;
ГБОУ ВО МО «Университет «Дубна»,
Институт системного анализа и управления;
141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: webstermaster777@gmail.com.

[2]Студент;
ГБОУ ВО МО «Университет «Дубна»,
Институт системного анализа и управления;
141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: kaliostrogoblin3@gmail.com.

[3]Профессор, доктор физико-математических наук;
ГБОУ ВО МО «Университет «Дубна»,
Институт системного анализа и управления;
141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: ososkov@jinr.ru.

Потеря урожая из-за болезней растений является серьезной проблемой для сельских жителей, экономики и продовольственной безопасности, требующей принятия своевременных мер для выявления и предотвращения болезней. В последнее время для решения задачи распознавания болезней растений по фотографиям их листьев стали с успехом применяться нейросетевые методы глубокого обучения. В настоящем исследовании выполнен анализ методов, используемых для обучения глубоких сверточных нейронных сетей в условиях малой обучающей выборки. Для данных PDD (http://pdd.jinr.ru/crops.php) применена техника переноса обучения и метода сиамских нейронных сетей с трехчленной функцией ошибки, что позволило достичь 99.5% точности классификации.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 18-07-00829.

Ключевые слова: распознавание, искусственные нейронные сети, глубокое обучение, перенос обучения.

THE CHOICE OF DEEP LEARNING METHODS FOR SOLVING THE PROBLEM OF RECOGNIZING PLANT DISEASES FOR CASES OF A SMALL TRAINING SAMPLE

Smetanin Artem[1], Goncharov Pavel[2], Ososkov Gennady[3]

[1]Student;
Dubna State University,
Institute of the system analysis and management;
141980, Dubna, Moscow reg., Universitetskaya str., 19;
e-mail: webstermaster777@gmail.com.

[1]Student;
Dubna State University,
Institute of the system analysis and management;
141980, Dubna, Moscow reg., Universitetskaya str., 19;
e-mail: kaliostrogoblin3@gmail.com.

Professor, Doctor of Science (Phys & Math);
Dubna State University,
Institute of the system analysis and management;
141980, Dubna, Moscow reg., Universitetskaya str., 19;
e-mail: ososkov@jinr.ru.

Loss of yield due to plant diseases is a serious problem for rural farmers, the economy and food security, requiring timely measures to identify and prevent diseases. Recently, neural network methods of deep penetration have been successfully applied to solve the problem of recognizing plant diseases from photographs of their leaves. This study analyzes the methods used to train deep convolutional neural networks for cases of a small training set. For the PDD data (http://pdd.jinr.ru/crops.php), the transfer learning technique and the Siamese neural network method with a three-term error function were applied, which allowed achieving 99.5% of the classification accuracy.

The reported study was funded by RFBR according to the research project № 18-07-00829.

Keywords: recognition, artificial neural networks, deep learning, transfer learning.

Введение

Для обнаружения и профилактики болезней сельскохозяйственных растений с использованием машинного обучения требуется не только разработать хорошую нейросетевую модель, но и создать программную систему, необходимую для комфортной работы с ней. Кроме того, в связи с ростом количества смартфонов, возникает потребность в создании мобильного приложения для организации интерактивного общения с пользователями. В ЛИТ ОИЯИ была разработана многофункциональная платформа *Plant Disease Detection Platform* (*PDDP*), которая дает возможность пользователям через веб-портал *pdd.jinr.ru* или мобильное приложение *PDDApp* отправлять фотографии и текстовые описания больных растений с целью определить заболевание и получить рекомендации по лечению культур и устранения причин возникновения заражения [1]. Для решения задачи распознавания болезней растений по фотографиям их листьев в *PDDP* с успехом используются технологии глубокого обучения [1, 2].

Наборы данных

Набор данных, предназначенных для обучения и тестирования нейронной сети, играет ключевую роль в применении нейросетевых методов. В открытом доступе по проблеме классификации болезней растений можно найти копию старой версии базы изображений проекта *PlantVillage* [3]. Набор состоит из 54 306 изображений листьев 14 культур (рис. 1), из которых 26 классов болезней и всего 38 категорий вместе со здоровыми листьями [4].

Рис. 1. Пример изображений из PlantVillage dataset

Была собрана собственная база изображений – *PDD dataset* [5]. Изображения были взяты из различных открытых источников (руководства, энциклопедии, форумы), все картинки приведены к

размеру 256x256, произведено центрирование. В наборе данных 611 изображений трех культур, таких как кукуруза, пшеница и виноград (рис. 2) – всего 15 категорий, из которых 12 классов болезней. В базе *PDD* представлены такие болезни как: Черная гниль, Апоплексия, Мучнистая роса, Хлороз (для винограда); Черный бактериоз, Бурная ржавчина, Мучнистая роса, Желтая ржавчина (для пшеницы); Склероспороз, Глазковая пятнистость, Бурая пятнистость, Ржавчина (для кукурузы). Также представлены изображения листьев здоровых растений для всех перечисленных культур.

Рис. 2. Примеры изображений из PDD dataset

В рамках исследований был реализован метод переноса обучения, используя для обучения данные из базы *PlantVillage*, что позволило добиться точности классификации на изображениях из этой же базы свыше 99%, однако та же программа при тестировании на изображениях из интернета показала точность менее 50% [6]. Главная проблема набора *PlantVillage* состоит в том, что в нем все изображения однотипные: один лист на сером фоне при равномерном освещении, что не характерно для фотографий, которые может сделать фермер на своем участке. Именно это и подтолкнуло к созданию собственного набора данных.

Сверточные нейронные сети

Для работы с изображениями на практике чаще всего используют глубокие сверточные нейронные сети (*convolutional neural networks, CNN*) – это искусственные нейронные сети, каждый слой которых состоит из нескольких операций, таких как операция свертки, функции активации и пулинга (субдискретизация).

Чтобы понять, как работает операция свертки, необходимо представить, что любое изображение – это матрица, или же сетка, из пикселей. Каждый пиксель имеет какое-то значение интенсивности цвета, а в случае общепринятых цветных изображений, три значения интенсивности для цветовой гаммы RGB (англ. red, green, blue – красный, зеленый, синий). Значения интенсивности находятся в пределах от 0 до 256. Размер изображения выражается в количестве пикселей по горизонтали и вертикали. Таким образом, цветное изображение размером 256x256, на самом деле, представляет собой тензор из 256x256x3 значений, т.е. 196608 значений интенсивностей пикселей, где 3 – это каналы цвета.

Идея операции свертки заключается в том, чтобы локально выделить набор признаков, которые будут полезны для дальнейшего распознавания. Поэтому для выполнения операции свертки, изображение разбивается на пересекающиеся регионы с размером региона, соответствующим размеру свертки. Степень пересечения регионов зависит от выбранного шага свертки. Матрица свертки для каждого признака – это матрица из весов, число строк в которой соответствует размеру свертки по вертикали, а число столбцов – размеру свертки по горизонтали. Сама же свертка – это линейное преобразование входных признаков путем скалярного перемножения вектора значений из входного окна и вектора весов свертки (рис. 3).

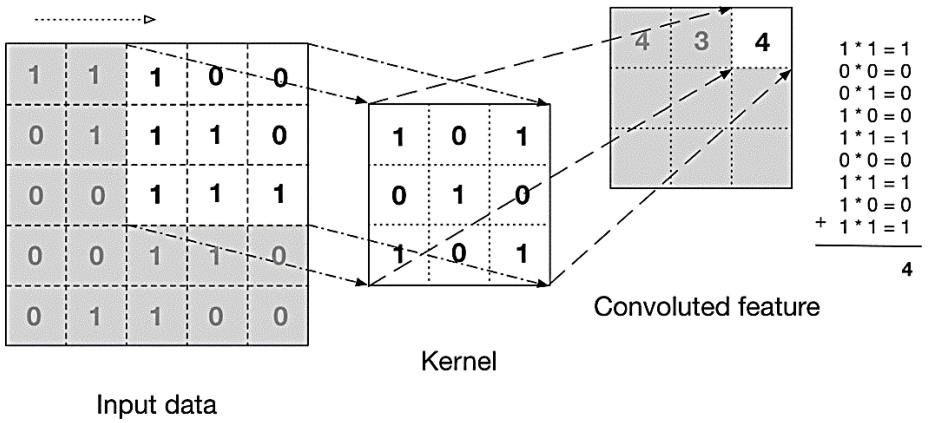

Рис. 3. Пример выполнения операции свертки

В каждом сверточном слое содержится не одна матрица весов для свертки, а множество матриц весов – это необходимо для того, чтобы каждая свертка выполняла роль локального детектора определенного признака. Какой признак будет извлекать свертка, заранее неизвестно и определяется процессом обучения сети. Число каналов на выходе определяется количеством извлекаемых признаков. Таким образом, если извлекать из изображения 128 различных признаков, то число каналов на выходе будет уже не 3, как в случае с *RGB*, а 128, и каждый канал отвечает за свой признак.

После выполнения операции свертки к полученному тензору применяется функция активации. Наиболее распространенной функцией активации в *CNN* является *ReLU*:

$$f(x) = \max(0, x). \quad (1)$$

Не все извлеченные операцией свертки признаки являются полезными для выполнения задачи, поэтому была придумана операция пулинга. Основная идея пулинга заключается в том, чтобы разбить карты признаков на пересекающиеся окна, как при свертке, и в каждом окне посчитать среднее или найти максимальное значение в зависимости от типа пулинга. Например, пулинг максимума (*max pooling*) – это извлечение максимального элемента, именно такой вид пулинга и применяется на практике чаще всего. Пулинг позволяет сжать изображение, выделив лишь самые значимые признаки. В результате применения операции пулинга будет получено изображение меньшего размера по сравнению с оригинальным входным изображением.

Более подробное описание архитектур сверточных сетей можно найти в [7].

Стратегии переноса обучения Transfer learning

На практике обучение глубоких *CNN* обычно не производится с нуля с произвольной инициализацией [8] – используется подход переноса обучения (*transfer learning – TL*). Необходимость применения *TL* в нашем случае обусловлена тем, что наш набор данных *PDD* не имеет достаточного размера, требуемого для обучения сети нужной глубины. Чаще всего, используют предварительно обученные на другом крупном наборе данных нейросети, потому что их веса являются хорошей стартовой позицией для инициализации обучения и такие сети быстрее сходятся к глобальному минимуму, кроме того, эти сети используют для выделения высокоуровневых признаков из исходных данных, не доучивая при этом саму сеть.

Можно выделить три основных этапа применения метода переноса обучения:

- выбирается глубокая нейросеть, предварительно обученная на крупном объеме данных;
- проводится точечная настройка выбранной модели путем запуска процедуры обучения на целевой выборке данных;
- выполняется оценка модели на тестовой части выборки и эксплуатация в рабочем режиме.

Глубокая сверточная сеть без применения *transfer learning*. Для того, чтобы сравнить, на сколько подход с переносом обучения эффективней, чем применение обычной нейронной сети, нами была создана и обучена модель *CNN*. В данной модели 5 сверточных блоков, включающих в себя:

операцию свертки; функцию активации *ReLU*; батч нормализацию [9]; и max pooling. Для обучения данной модели использовались следующие параметры: набор данных – *PDD dataset;* оптимизатор – стохастический градиентный спуск (*SGD*) со скоростью обучения 0.001 и моментом 0.9; функция потерь – кросс-энтропия; размер батча – 16 изображений; количество эпох обучения – 100. Эффективность классификации данной модели составила 61%, что означает, что обученная модель при тестировании допустила ошибки в 39% случаев. Такая точность классификации является неудовлетворительной.

Глубокая сверточная сеть с применением *transfer learning.* Стратегии переноса обучения зависят от разных факторов, но наиболее важными являются два: размер целевого набора данных и его схожесть с исходным набором данных. Характер работы глубокой сверточной сети более универсален на ранних слоях и становится более тесно связанным с конкретным набором данных на последующих слоях. Одной из проблем является выбор подходящей модели для переноса обучения. Для сравнения были выбраны две модели: одна, основанная на архитектуре *ResNet50* [10], а вторая – *MobileNetV2* [11]. Выбор в качестве второй модели, предназначенной для мобильных устройств, мотивирован дальнейшими планами по выполнению классификации в режиме оффлайн, т.е. прямо на пользовательском устройстве, без подключения к сети Интернет.

Обе модели для переноса обучения были натренированы с такими же параметрами, как и обученная с нуля *CNN*. Точность классификации *ResNet*50 составила 77%, что уже значительно лучше обычной глубокой сверточной сети. Точность классификации *MobileNetV2* на тестовых данных составила 87%, что, в частности объяснимо тем фактом, что модель для мобильных устройств меньше, следовательно, ее обучение на небольшой обучающей выборке PDD идет более эффективно, без опасности переобучения .

Чтобы улучшить результат классификации, некоторые настройки были изменены: все веса модели заморожены – обучаемый только последний слой-классификатор, в котором число выходных нейронов соответствует количеству классов в целевом датасете, а именно 15. Чтобы искусственно увеличить размер тренировочной выборки, к изображениям были применены случайные повороты изображений по горизонтали и вертикали. Ко всем изображениям была применена нормализация такая же как у изображений *PDD dataset*: использовались параметры среднего по цветовым каналам – [0.4352, 0.5103, 0.2836], и девиация – [0.2193, 0.2073, 0.2047]. Также был увеличена скорость обучения у оптимизатора с 0.001 до 0.006. Такие изменения позволили поднять точность классификации до 89%.

One-shot learning и сиамские нейросети

Применение переноса обучения позволило получить результат лучше, чем обучение с нуля, но все равно недостаточный, чтобы использовать такую модель в реальном приложении, поскольку сети, использованные для *TL*, были обучены для решения задачи классификации на своей базе, сильно отличной от целевого домена проблемы распознавания болезней растений, а новых тренировочных примеров оказалось недостаточно для доучивания. Признаки, которые научились извлекать TL сети для того, чтобы отличать различные виды предметов и виды животных, не подходят для определения культуры растения по фотографии листа и идентификации болезни по наличию мелких пятен заражения, а также их цвету. Для того, чтобы сеть умела распознавать необходимые нам признаки, требуется сделать обучаемой большую часть нейронных слоев, оставив нетронутыми лишь начальные слои, которые реагируют на границы предметов и градиенты, но такая «разморозка» все равно требует сотен тысяч изображений больных и здоровых листьев.

Существуют методы, которые позволяют обучить сеть, используя экстремально малый объем данных для обучения. Их называют обучением с одного «выстрела» (*one-short learning*) [12]. Основная идея *one-shot* обучения заключается в том, чтобы подготовить модель, используя при этом всего один или несколько тренировочных примеров на каждую категорию в данных. Сиамские сети являются частным случаем *one-shot* обучения [12].

Сиамская сеть состоит из двух сетей-близнецов, соединенных между собой слоем подобия. Веса близнецов связаны (одинаковы), поэтому результат является инвариантным и, кроме того, гарантирует, что очень похожие изображения не могут находиться в очень разных местах в пространстве объектов. Слой подобия определяет некоторую метрику расстояния между так

называемыми вложениями, высокоуровневыми представлениями характеристик входной пары изображений. Обучение такой сети происходит на парах изображений: на вход каждого близнеца подается обучающий пример; примеры могут быть выбраны из одной или из разных категорий; задача модели сделать так, чтобы вектора признаков изображений из одной категории в процессе обучения становились как можно ближе друг к другу, а вектора вложений примеров из разных классов отстояли как можно дальше друг от друга. Таким образом, сеть учится различать изображения по степени «похожести». На рисунке 4 представлена схема работы сиамской нейронной сети.

Тренировка по парам также является более выгодной, поскольку число создаваемых пар изображений для обучения модели возрастает квадратично, что препятствует переобучению нейросети.

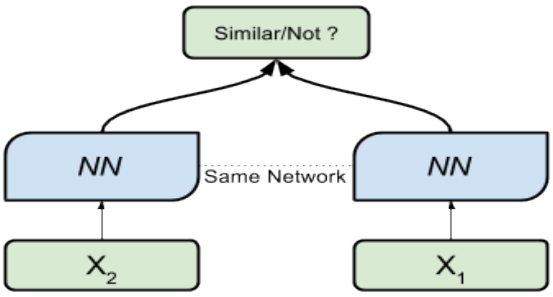

Рис. 4. Пример сиамской сети с использованием изображений из PDD dataset. Два блока NN – сети близнецы; x_1 и x_2 – пара входных изображений; на выходе блок, представляющий собой слой подобия

Этапы применения и общая схема работы с сиамской сетью:

- обучить сиамскую сеть;
- получить признаки высокого уровня, используя обученную на предыдущем этапе сеть-близнец;
- классифицировать, используя алгоритм ближайшего соседа.

Авторы [2] обнаружили, что для всех 15 классов обучение сиамской сети с нуля и использование KNN в качестве классификатора становится не эффективным. Точность классификации на тестовой выборке PDD датасета достигла 86%, однако, заменив KNN на многослойный персептрон, удалось улучшить качество распознавания до ~95%.

Трехчленная функция потерь в сиамских сетях

Для того, чтобы улучшить качество классификации и сделать классы более разделимыми в пространстве, в данной работе рассматривается метод с использованием специальной трехчленной функции потерь (триплет-ошибка), такая же, как использовалась в модели *FaceNet* от *Google* [13].

Поскольку мы хотим сравнить два изображения и получать небольшие расстояния для одинаковых изображений и большие расстояния для разных изображений (рис. 5), идея состоит в том, чтобы рассмотреть сразу тройку изображений (триплет) (рис. 6):

- начальное изображение, называемое «якорем» *(anchor*);
- изображение того же класса что и «якорь» (*positive*);
- изображение другого класса (*negative*).

Трехчленная функция ошибки минимизирует расстояние между *anchor* и *positive*, так как оба они принадлежат одному классу, и максимизирует расстояние между *anchor* и *negative*, иллюстрация этого процесса продемонстрирована на рисунке 5. На рисунке 6 представлен пример составленного триплета из базы изображений проекта *PDDP*.

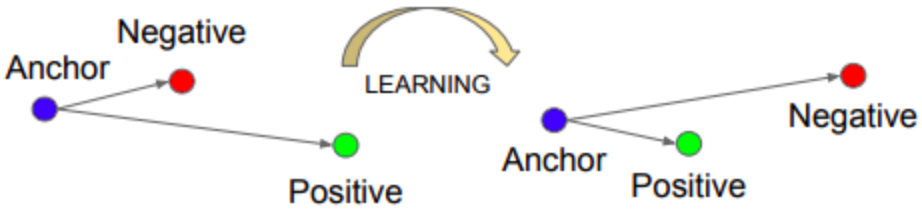

Рис. 5. Обучение сети с трехчленной функцией ошибки [13]. Слева триплет до итерации обучения, а справа уже скорректированы расстояния после обучения.

Формула трехчленной функции ошибки:

$$L = \max(d(a,p) - d(a,n) + margin, 0), \quad (2)$$

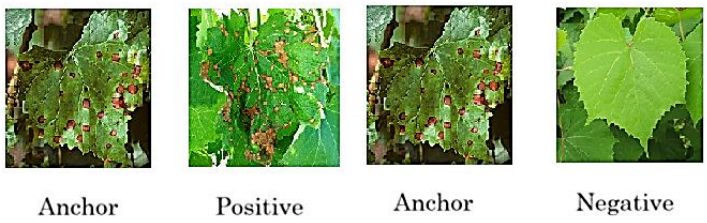

Рис. 6. Примеры изображений Anchor, Positive, Negative

где переменная «*a*» представляет собой якорное изображение, «*p*» – изображение из того же класса, что и якорное, а «*n*» представляет собой отрицательное изображение. Формула определяет, что различие между *anchor* и *positive* должно быть меньше, чем различие между *anchor* и *negative*, степень этого различия определяется параметром *margin*. В уравнении 2 d – это некая функция для подсчета расстояния между векторами, например, Евклидовое расстояние.

Трехчленная функция ошибки для набора данных *PDD*. Авторы в [13] показали, что для обучения с помощью триплет-ошибки лучше использовать предобученную сеть. Нами за основу была взята предобученная на наборе *ImageNet* сеть *MobileNetV*2. Т. е. берется уже натренированная с помощью кросс-энтропии на большом объеме данных модель; все веса делают обучаемыми; сеть ложится в основу одного из близнецов сиамской триплет-сети, и уже сформированная таким образом сиамская сеть доучивается с помощью триплет-ошибки. После того, как сиамская триплет-сеть обучена, веса одного из близнецов используются для извлечения высокоуровневых признаков из исходных изображений с целью дальнейшей классификации методом ближайшего соседа.

Для генерации триплетов использовалась библиотека *TripletTorch* [14], а именно, класс *AllTripletMiner* с параметром *margin* равным 0.5. Сеть обучалась три тысячи эпох (рис. 7, 8) с батчем 32. Данные базы *PDD* разбивались на тренировочную и тестовую выборки в пропорциях 8 к 2. Сиамская триплет-сеть обучалась на тренировочном наборе данных, а тестировалась на тестовой подвыборке. Чтобы не подстраиваться под одно конкретное разбиение, данные случайным образом делились 5 раз на обучающую выборку и тест. В результате обучения для классификатора *KNN* для одного и трех соседей удалось достичь точности классификации 99,5% на тестовой выборке в одном из пяти разбиений и в среднем точность классификации составила 97%.

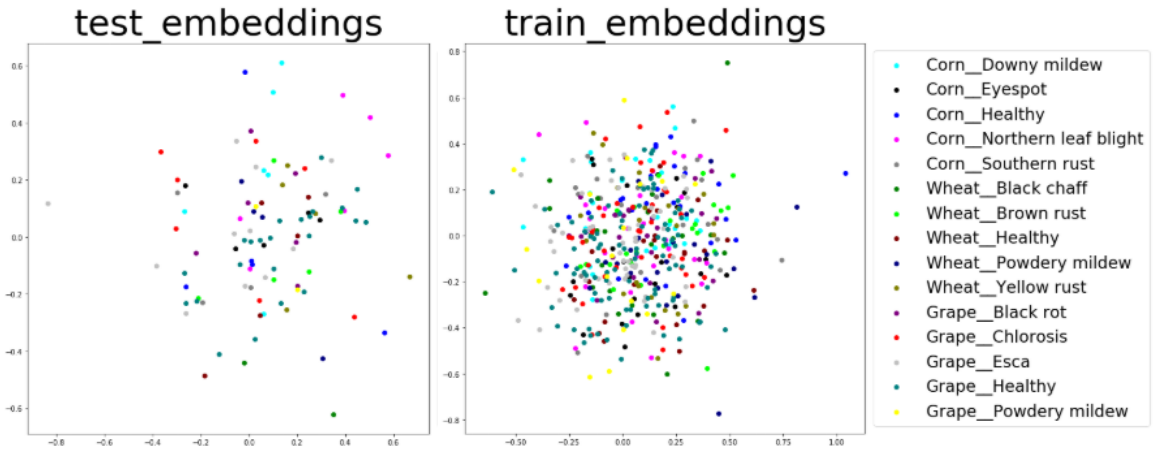

Рис. 7. Расположение классов в двумерном пространстве, до обучения сети.

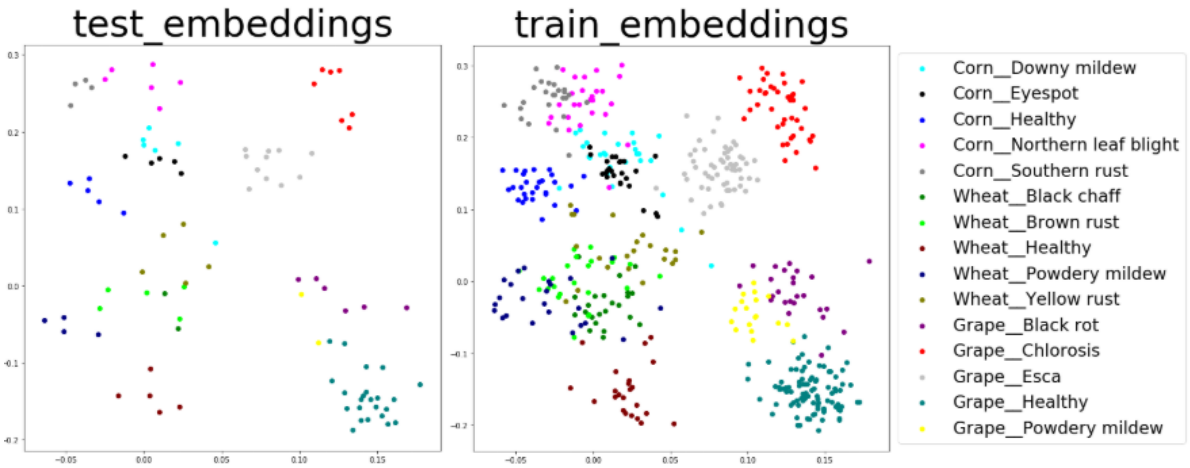

Рис. 8. Расположение классов в двумерном пространстве, после обучения сети

На рисунках 7 и 8 изображено распределение тренировочных и тестовых примеров до (рис. 7) и после обучения (рис. 8). Изображения слева – это распределение тренировочных данных, а справа – распределение тестовых примеров данных. Разные цвета означают принадлежность к разным классам. Рисунки получены путем проекции векторов скрытых признаков, извлеченных обученной триплет-сетью на двумерную плоскость. Даже несмотря на сильное сжатие данных (с 1280 признаков до 2), можно увидеть, что классы очень хорошо разделены в пространстве, в то время как до обучения (рис. 7) все точки распределены на плоскости равномерно. Близко друг от друга находятся изображения, принадлежащие одной культуре, например, это хорошо заметно, как сгруппированы точки, относящиеся к пшенице на рисунке 8. График на рисунке 8 свидетельствует о том, что сеть хорошо обучилась разделять изображения разных категорий в пространстве скрытых признаков, что также подтверждается высокой точностью распознавания метода *KNN*.

Заключение

В данном исследовании был проведен анализ применения методов глубокого обучения, используемых для обучения глубоких нейронных сетей в условиях малой обучающей выборки.Было обучено три глубоких сверточных нейросети на данных *PDD* датасета с использованием кросс-энтропии в качестве ошибки классификации: CNN с пятью сверточными блоками, *ResNet50* и *MobileNetV2*, точность классификации на тестовой подвыборке данных *PDD* составила 61%, 77% и 87%, соответственно.

Для улучшения результата классификации был использован метод переноса обучения, в качестве предобученной модели использовалась *MobileNetV2*. Точность классификации составила 89% на тестовой подвыборке.

Чтобы справиться с проблемой малой обучающей выборки был выбран метод сиамских нейронных сетей с триплет-ошибкой. В качестве базовой сети использовалась предобученная *MobileNetV2*. Лучший результат классификации на тестовой выборке для глубокой сверточной нейронной сети с применением переноса обучения с трехчленной функцией ошибки составил 99.5% точности.

Программный код модели, составление датасета и обучения находится в открытом доступе по ссылке [15]. Все вычисления производились с помощью экосистемы для машинного обучения на базе платформы *HybriLIT* Объединенного института ядерных исследований [16].

Список литературы:

1. Uzhinskiy A. et al. Multifunctional Platform and Mobile Application for Plant Disease Detection / A. Uzhinskiy, G. Ososkov, P. Goncharov, A.Nechaevskiy //Proceedings of the 27th Symposium on Nuclear Electronics and Computing (NEC 2019), Budva, Montenegro. — 2019. — Vol. 2507. — P. 110-114.
2. Goncharov P. et al. Deep Siamese Networks for Plant Disease Detection / Pavel Goncharov, Alexander Uzhinskiy, Gennady Ososkov, Andrey Nechaevskiy and Julia Zudikhina // EPJ Web of Conferences. — EDP Sciences, 2020. — Vol. 226. — P. 03010.
3. PlantVillage главная страница. — [Электронный ресурс]: https://plantvillage.psu.edu/. (Дата использования 3.11.2019).
4. Dataset of diseased plant leaf images and corresponding labels. — [Электронный ресурс]: https://github.com/spMohanty/PlantVillage-Dataset (Дата использования 3.11.2019).
5. Наборы изображений PDDP. — [Электронный ресурс]: http://pdd.jinr.ru/crops.php (Дата использования 3.11.2019).
6. Goncharov P. et al. Disease Detection on the Plant Leaves by Deep Learning / P. Goncharov, G. Ososkov, A. Nechaevskiy, A. Uzhinskiy, I. Nestsiarenia // Advances in Neural Computation, Machine Learning, and Cognitive Research II: Selected Papers from the XX International Conference on Neuroinformatics, October 8-12, 2018, Moscow, Russia. — Springer, 2018. — Vol. 799. — P. 151.
7. Кадурин А. Глубокое обучение. Погружение в мир нейронных сетей / А. Кадурин, Е. Архангельская, С. Николенко. — СПБ.: Питер. — 2018. — С. 480.
8. Torrey L., Shavlik J. Transfer learning // Handbook of research on machine learning applications and trends: algorithms, methods, and techniques. — IGI Global, 2010. — Pp. 242-264.
9. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift //arXiv preprint arXiv:1502.03167. — 2015.
10. He K. et al. Deep residual learning for image recognition //Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — Pp. 770-778.
11. Sandler M. et al. Mobilenetv2: Inverted residuals and linear bottlenecks //Proceedings of the IEEE conference on computer vision and pattern recognition. — 2018. — Pp. 4510-4520.
12. Koch G. et al. Siamese neural networks for one-shot image recognition / G. Koch, R. Zemel, R. Salakhutdinov//ICML deep learning workshop. — 2015. — Vol. 2.
13. Schroff F., Kalenichenko D., Philbin J. Facenet: A unified embedding for face recognition and clustering //Proceedings of the IEEE conference on computer vision and pattern recognition. — 2015. — Pp. 815-823.
14. Triplet Loss Utility for Pytorch Library. — [Электронный ресурс]: https://github.com/TowardHumanizedInteraction/TripletTorch (Дата использования 5.11.2019).
15. Репозиторий Plant Disease Detection. — [Электронный ресурс]: https://github.com/WEBSTERMASTER777/pdd (Дата использования 4.02.2020).
16. Web-portal of HybriLIT JINR computation facility [Electronic resource]. — Mode of access: http://hybrilit.jinr.ru. — Date of access: 5.11.2019.