МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Пермский государственный национальный исследовательский университет»

УДК 004.852

Кафедра математического обеспечения вычислительных систем

Байесовская статистика для повышения качества классификации редких классов изображений

Курсовая работа

Работу выполнил
студент группы ПМИ-1,2-2019 НБ
3 курса механико-математического факультета
Проскуряков Кирилл Александрович
«___» _____2022 г.

Научный руководитель:
к.т.н., PhD,
доцент кафедры МОВС
Бузмаков Алексей Владимирович
«___» _____2022 г.

Пермь 2022

Содержание

Введение 3

Глава 1. Обзор сравниваемых моделей для классификации 6

1.1. Обзор свёрточных нейронных сетей 6

1.1.1. Архитектура свёрточной нейронной сети 7

1.1.2. Анализ некоторых моделей CHC 13

1.1.3. Анализ EfficientNetV2 15

1.2. Обзор методов байесовской статистики 18

1.2.1. Байесовский подход в машинном обучении 18

1.2.2. Байесовская логистическая регрессия 19

Глава 2. Реализация описанных моделей для классификации 21

2.1. Описание выбранного набора данных 21

2.2. Разбиение исходного набора данных 21

2.3. Создание модели свёрточной нейронной сети 22

2.4. Обучение и тестирование свёрточной нейронной сети 23

2.5. Реализация байесовской логистической регрессии 26

2.5.1. Подготовка входных данных для модели 26

2.5.2. Создание модели 26

Глава 3. Анализ результатов классификации 28

3.1. Анализ классификации класса Grape leafhopper 28

3.2. Анализ классификации класса Laurel leaf weevils 36

3.3. Выводы 38

Заключение 39

Список использованных источников 40

Введение

Классификация изображений является одной из важнейших задач компьютерного зрения – области искусственного интеллекта, позволяющей компьютерам извлекать полезную информацию из изображений, видео и других визуальных данных и предпринимать действия или давать рекомендации на основе этой информации [1].

Сущность задачи классификации заключается в определении, к какому из заранее определённого конечного набора классов принадлежит объект.

Значимость данной задачи сложно переоценить – она решается повсеместно во многих сферах нашей жизни: от поиска породы понравившейся нам кошки в Google Lens [2] или аналогичном сервисе, до обнаружения запрещённого контента при загрузке изображения в социальной сети и диагностирования заболеваний в медицине. Помимо этого, классификация является основой для решения более сложных задач компьютерного зрения – локализации, сегментации и детекции [3].

В настоящее время для решения задачи классификации изображений чаще всего используются свёрточные нейронные сети (СНС) – искусственные нейронные сети (ИНС) специального вида, ориентированные на работу с растровыми изображениями. Однако, качество классификации во многом определяется количеством и качеством данных, используемых для обучения нейронной сети. В связи с этим, часто возникает проблема низкого качества классификации редких классов, то есть классов, представленных сравнительно небольшим количеством примеров в обучающем множестве. Действительно, поскольку на практике часто бывают случаи, когда не удаётся собрать достаточно примеров для обучающего множества, например, когда нет общедоступных наборов данных, а собирать и размечать данные самостоятельно слишком затратно.

Существуют различные методики повышения качества классификации редких классов, одной из которых является применение методов, основанных на байесовской статистике.

В байесовском подходе, в отличие от частотного, параметры оцениваются не точечно, а плотностью их распределения. Для оцениваемого параметра задаётся априорное распределение его плотности, в которое можно вложить знания о том, как он может быть распределён. Далее, при получении новых данных, посредством применения теоремы Байеса, это распределение обновляется и становится априорным уже для следующих полученных данных.

Поэтому, применяя байесовские методы, помимо самого предсказания класса, мы также получаем оценку неопределённости и уменьшаем её с ростом числа наблюдений.

Таким образом, можно выдвинуть *гипотезу*, что байесовское методы могут помочь увеличить качество предсказания редких классов изображений.

Объектом исследования будет являться задача классификации изображений, а *предметом исследования* – байесовская статистика для повышения качества классификации редких классов.

Целью курсовой работы является создание свёрточной нейронной сети, а также байесовской статистической модели для классификации изображений и сравнение их результатов при классификации редких классов для проверки выдвинутой гипотезы.

Для достижения поставленной цели требуется решить следующие *задачи*:

1. Провести анализ современных моделей СНС и выбрать оптимальный вариант по соотношению качества классификации и скорости обучения.
2. Обучить и протестировать выбранную модель на предоставленном наборе данных.
3. Выбрать байесовскую модель для классификации и реализовать её.

4. Сравнить результаты классификации редких классов, полученные СНС и байесовской моделью.

Глава 1. Обзор сравниваемых моделей для классификации

В данной главе обозреваются теоретические основы свёрточных нейронных сетей, а также выбирается оптимальная модель для выполнения практической части работы. Помимо этого, в ней описываются основы байесовской статистики и её применение в задаче классификации, выбирается оптимальная байесовская статистическая модель для реализации на практике.

1.1. Обзор свёрточных нейронных сетей

В настоящее время для решения задачи классификации объектов на изображениях часто используются свёрточные нейронные сети – искусственные нейронные сети специальной архитектуры, ориентированные на эффективную работу с растровыми изображениями. Популярность СНС объясняется явным превосходством их результатов над результатами классических методов во многих задачах компьютерного зрения.

Архитектура свёрточных нейронных сетей была вдохновлена биологическими процессами, происходящими в зрительной коре животных. Эти процессы исследовались в 60-е годы прошлого века нейрофизиологами Дэвидом Хьюбелем (*David H. Hubel*) и Торстеном Визелем (*Torsten Wiesel*) [4,5]. Учёные подключали микроэлектроды в первичную зрительную кору анестезированных животных и показывали им изображения под разными углами. Выяснилось, что некоторые нейроны зрительной коры быстро активировались при наблюдении за линиями под одним углом, в то время как другие нейроны лучше реагировали на линии под другими углами. Некоторые нейроны активировались на светлые или тёмные узоры, пока другие реагировали при движении линий в определённом направлении. Эти нейроны называются простыми клетками. Помимо простых клеток, также были открыты и сложные, реакция которых связана с активацией определённого набора простых клеток.

Основы архитектуры современных СНС были заложены сотрудником Bell Labs Яном Лекуном (*Yann LeCun*) в 1989 году [6]. Исследователь был

первым, кто создал свёрточную нейронную сеть и применил в ней для обучения алгоритм обратного распространения ошибки. Данная СНС получила название LetNet-5 и была успешно применена для идентификации рукописных номеров почтовых индексов, предоставленных почтовой службой США.

Архитектура LetNet-5 представлена на рисунке 1.

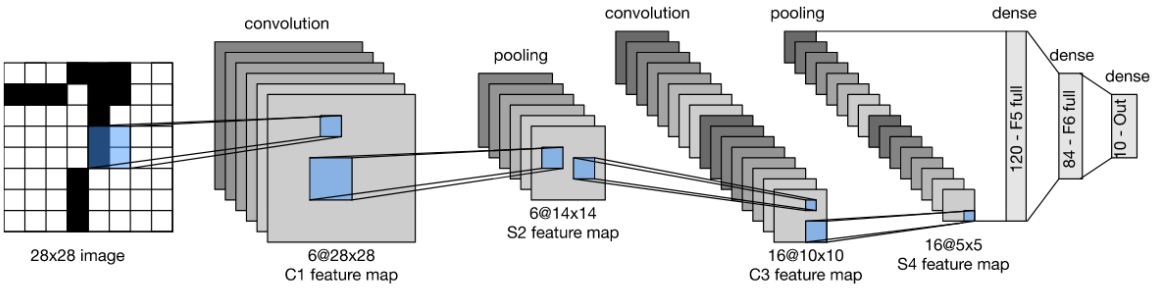

Рисунок 1. Архитектура LeNet-5 [7]

1.1.1. Архитектура свёрточной нейронной сети

Архитектура свёрточной нейронной сети является многослойной и однонаправленной (в ней отсутствуют обратные связи). Она состоит из входного слоя, выходного слоя, а также скрытых слоёв, находящихся между ними, в которых и происходят все вычисления.

В архитектуре СНС выделяют три основных вида скрытых слоёв [8]:

- Слой свёртки – слой, основанный на операции обработки изображений, называемой свёрткой. Суть данной операции заключается в том, что входное изображение фильтруется с помощью небольшой матрицы, называемой ядром (фильтром) свёртки.
- Слой пуллинга (субдискретизации) – слой, призванный уменьшить размерность входного изображения. Как правило, следует за слоем свёртки.
- Полносвязные слой – слой, находящийся в конце сети и используемый в качестве классификатора для получения оценок принадлежности к каждому из классов.

1.1.1.1. Слой свёртки

Слой свёртки (англ. *Convolutional Layer*) является основным блоком для построения СНС. Он используется для выявления признаков на входном изображении. Слой представляет собой набор обучаемых ядер свёртки (англ. *Kernels*). Обычно ядра имеют сравнительно небольшой размер, например 3x3 (то есть 3 пикселя в длину и 3 пикселя в ширину).

Операция свёртки заключается в последовательном проходе ядра по матрице входного изображения, начиная с левого верхнего угла, и заканчивая правым нижним. При этом проходе происходит поэлементное умножение проходимых элементов матрицы входного изображения на элементы ядра, далее полученные значения суммируются и записываются в соответствующий элемент результирующей матрицы, называемой картой признаков (англ. *Feature map*).

Процесс свёртки изображён на рисунке 2.

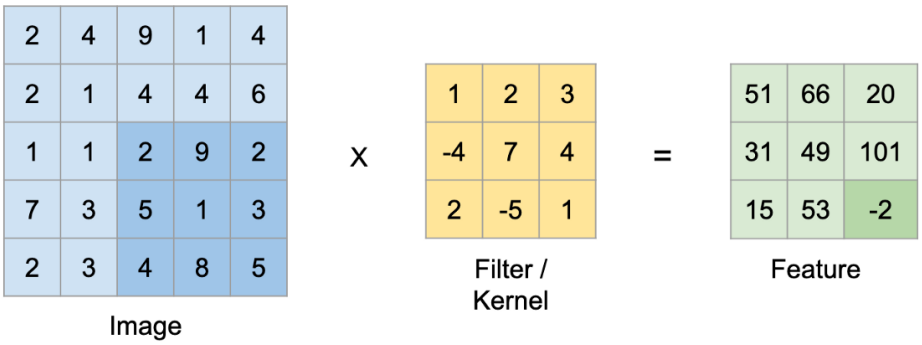

Рисунок 2. Свёртка входного изображения 5x5 ядром размера 3x3

Как можно заметить, размерность карты признаков получается меньше размерности входного изображения. Для манипуляции размерами карт признаков используются два параметра:

- Дополнение (англ. *Padding*) – добавление пикселей к краям исходного изображениями. Обычно изображение дополняется нулями. При использовании дополнения, размер карты признаков совпадёт с размером исходного изображения (при единичном шаге).

- Шаг (англ. *Stride*) – шаг, с которым ядро будет проходить по матрице исходного изображения. Например, при использовании шага 2, размер карты признаков на рисунке 2 составил бы 2x2.

Каждое ядро отвечает за выявление какого-либо признака на изображении, например, вертикальных или горизонтальных линий. Чем более явно на изображении выражен признак, тем больше будут значения в соответствующей карте признаков. Чем «глубже» находится свёрточный слой в сети, тем более высокоуровневые признаки будут обнаруживать его ядра.

Повышение уровня признаков, обнаруживаемых свёрточным слоем в зависимости от его местонахождения, изображено на рисунке 3.

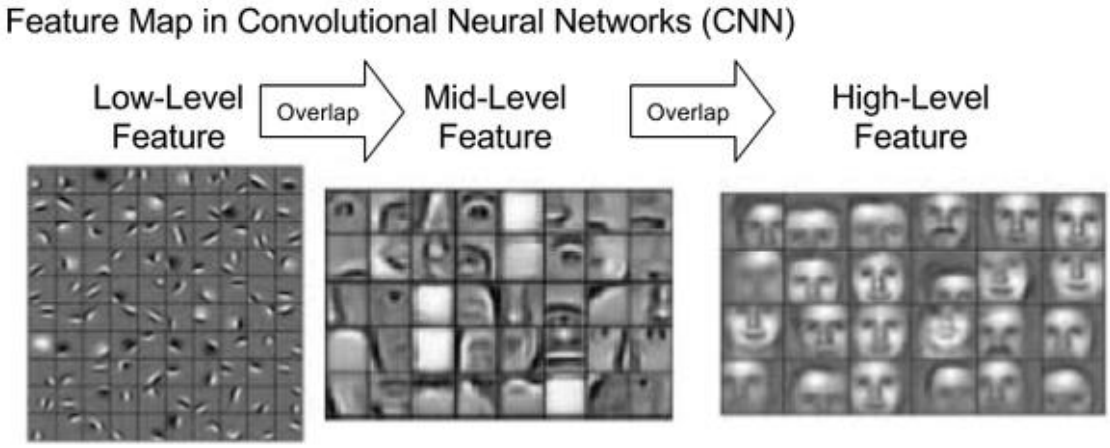

Рисунок 3. Повышение уровня признаков, обнаруживаемых свёрточным слоем

Стоит отметить, что полученные карты признаков в действительности не являются изображениями, а рисунок был приведён для наглядности.

Для того, чтобы при свёртке могли выявляться нелинейные признаки на входном изображении, необходимо после получения карты признаков поэлементно применить к ней нелинейную функцию активации (англ. *Activation Function*). Традиционно в качестве таких функций выступали гиперболический тангенс и сигмоида [9]. Однако эти функции обладают двумя серьёзными недостатками:

- Вычислительная сложность. Поскольку при обучении СНС значение функции и её градиент вычисляются многократно, сложность их вычисления значительно увеличивает время обучения.

- Затухающий градиент. В ходе обучения СНС может возникнуть ситуация, когда градиент станет настолько мал, что процесс обучения будет проходить очень медленно, либо остановится вовсе.

В настоящее время разработчиками и исследователями активно используется функция активация ReLU (Rectified Linear Unit), популяризированная в [10]. Эта функция активации, заменяющая отрицательные значения нулями, позволила значительно ускорить процесс обучения за счёт сокращения объёма вычислений. Также ReLU, за счёт того, что её производная при положительных значениях аргумента всегда равна единице менее подвержена проблеме затухающего градиента.

График функции ReLU представлен на рисунке 4.

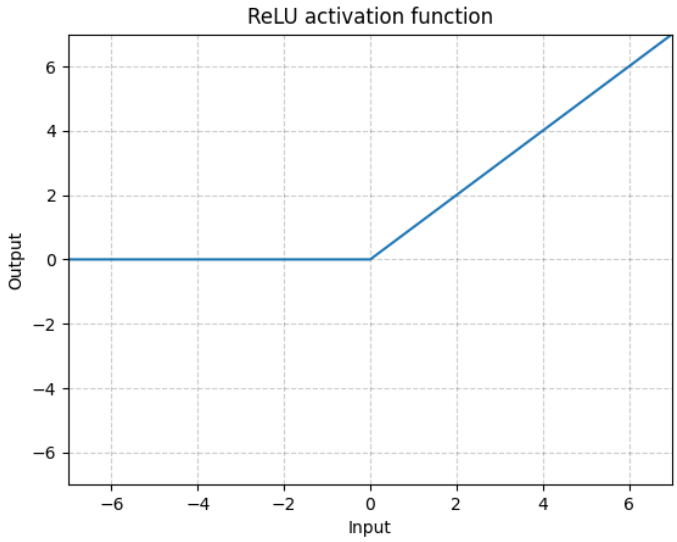

Рисунок 4. Функция активации ReLU

1.1.1.2. Слой пуллинга

Слой пуллинга (англ. *Pooling Layer*) используется для уменьшения размерности карты признаков. В ходе работы данного слоя происходит значительное сжатие информации, поступающей на вход. Однако, если на свёрточном слое был выявлен какой-то признак, то для его дальнейшей обработки нет необходимости хранить всю карту признаков и её можно сжать.

К главным причинам использования слоя пуллинга можно отнести:

- Повышение устойчивости СНС к небольшим сдвигам и поворотам объектов на изображениях.

- Повышение скорости обучения. Поскольку карты признаков являются входными данными для следующих свёрточных или полносвязных слоёв, уменьшение их размерности позитивно сказывается на скорости обучения.

Сама операция пуллинга представляет собой уплотнение карты признаков следующим образом: к непересекающимся группам элементов прямоугольной или квадратной формы (на практике обычно выбирают группы размерности 2x2.) применяется функция, ставящая в соответствие этой группе число.

Изначально в роли такой функции выступало взятие среднего значения элементов (*Average Pooling*), однако в настоящее время стандартным решением является выбор максимального значения в группе элементов (*Max Pooling*). На практике обнаружилось, что для задач компьютерного зрения применение именно функции выбора максимального значения из группы элементов даёт лучшие результаты.

Результаты работы слоя пуллинга с функциями выбора максимального и среднего значений из группы представлены на рисунке 5.

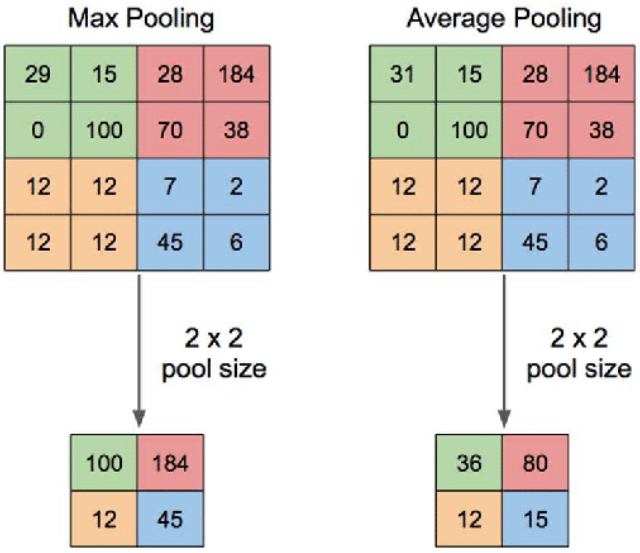

Рисунок 5. Результат работы наиболее популярных слоёв пуллинга

В классической архитектуре СНС слои пуллинга следуют сразу же за слоями свёртки, объединяясь таким образом в блоки. Такой блок получает на вход изображение, обрабатывает его фильтрами (обычно размерности 3x3) на

слое свёртки, получая карты признаков, далее применяет к ним нелинейную функцию активации (как правило, ReLU) и передаёт на вход слою пуллинга. В слое пуллинга полученные карты активации делятся на группы (обычно группа имеет размерность 2x2), далее к каждой группе применяется функция, ставящая ей в соответствие число (зачастую выбор максимального значения из группы), таким образом понижая размерность карт активаций. На выходе блок выдаёт набор карт активаций с пониженной размерностью.

1.1.1.3. Полносвязный слой

В результате последовательной обработки входного изображениями блоками свёрток и пуллингов, на выходе получается набор высокоуровневых карт признаков, выделенных из входного изображения.

На следующем этапе работы СНС все данные полученных карт признаков конкатенируются в один вектор, поступающий на вход полносвязный нейронной сети. Обычно такая нейронная сеть состоит из одного или нескольких слоёв. Как следует из названия, в полносвязном слое каждый нейрон текущего слоя связан с каждым нейроном следующего слоя. Обычно в качестве функции активации после полносвязных слоёв используется ReLU. При решении задачи небинарной классификации, к выходам последнего полносвязного слоя применяется функция Softmax, нормализующая выходные значения и выдающая вероятности принадлежности изображения к классам.

1.1.1.4. Процесс обучения

Процесс обучения ИНС заключается в изменении параметров модели таким образом, чтобы выходные значения были близки к желаемым.

В рассмотренной классической архитектуре СНС параметрами модели будут являться:

- Значения ядер и пороговые значения (по одному на ядро), прибавляемые ко всем элементам карты признаков в свёрточных слоях.

- Значения весов и пороговые значения (по одному на нейрон) в полносвязных слоях.

Именно благодаря тому, что значения ядер являются параметрами модели, СНС обучается находить и извлекать необходимые признаки из изображений.

Для определения «близости» желаемых и выходных значений используется функция потерь (англ. *Loss Function*). При решении задачи небинарной классификации в качестве функции потерь обычно используется перекрёстная энтропия (англ. *Cross Entropy*). Таким образом, значения параметров модели на каждой эпохе обучения корректируются так, чтобы уменьшить значение функции потерь.

Наиболее популярным алгоритмом обучения ИНС является обратное распространение ошибки [11]. На каждой эпохе обучения происходит два прохода по сети – прямой и обратный. Во время прямого прохода входные данные проходят по сети и формируют предсказание выходных данных. Далее предсказанные выходные данные сравниваются с фактическими и считается значение функции потерь. На обратном проходе находится градиент функции потерь и распространяется от выхода сети к её входу. Зная значение градиента функции потерь на следующем слое, по правилу дифференцирования сложной функции можно посчитать градиент текущего, и изменить значение параметров в обратном от градиента направление. Таким образом, алгоритм использует так называемый стохастический градиентный спуск, «продвигаясь» в многомерном пространстве значений параметров в направлении антиградиента с целью минимизации функции потерь.

1.1.2. Анализ некоторых моделей СНС

В последнее десятилетие, в связи с ростом вычислительных мощностей, появлением новых алгоритмов и созданием больших наборов данных стало возможно эффективно обучать глубокие свёрточные нейронные сети, то есть сети, состоящие из десятков и сотен слоёв. Так, каждый год, начиная с 2012, соревнование ImageNet Large Scale Visual Recognition Challenge (ILSVRC) выигрывали исключительно участники, использовавшие свёрточные

нейронные сети [12]. Сам ImageNet является набором данных, изначально включавшим в себя 1000 классов и около миллиона изображений, но постоянно пополняющимся силами сообщества.

В 2012 году на ILSVRC победила СНС AlexNet [13]. В ней использовались функция активации ReLU и слои пуллинга с выбором максимального значения из группы элементов, а сама модель обучалась на графическом процессоре, позволившем значительно сократить время обучения. Ошибка классификации на тестирующем множестве составила всего 15.3%, что оказалось на 10.8% ниже, чем у ближайшего преследователя. Именно после победы AlexNet многие исследователи и компании заинтересовались в разработке и внедрении всё более совершенных архитектур СНС.

Так, в 2014 году на соревновании ILSVRC победила модель VGG [14]. Данная модель была разработана Оксфордским университетом для распознавания объектов на изображениях. У этой модели имеется четыре варианта: VGG-11, VGG-13, VGG-16 и VGG-19, где номер 11, 13, 16 и 19 указывают на количество слоёв в сети. Точность модели VGG-16 составила 92.7%. Главный вывод, который сделали авторы данной модели заключался в том, что эффективно работы СНС напрямую зависит от количества скрытых слоёв в ней.

В 2015 году Microsoft Research Asia разработали архитектуру ResNet [15]. Ключевой особенностью этой архитектуры является наличие остаточных соединений (англ. *Residual Connection*) – соединяющих выходы $(I - n)$ – го слоя со входом I – го слоя. Наличие таких соединений помогает бороться с проблемой затухающего градиента, что в свою очередь позволяет создавать и эффективно обучать глубокие СНС. Пример остаточного соединения схематично представлен на рисунке 6.

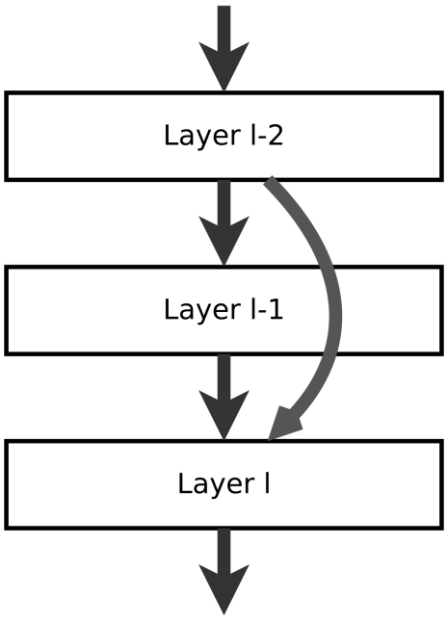

Рисунок 6. Пример остаточного соединения

Помимо этого, ещё одной особенностью архитектуры является то, что её выходной слой – это не полносвязный слой, а слой пуллинга, после которого идёт функция Softmax.

В данном разделе было описано всего несколько архитектур СНС из всего их огромного разнообразия для того, чтобы показать, что несмотря на достаточно высокую эффективность уже существующих моделей, постоянно создаются всё новые, более совершенные. Однако все архитектуры в той или иной степени сталкиваются с одной из важнейших проблем современных СНС – длительным временем их обучения. Так, современная СНС на достаточно большом наборе данных может обучаться днями, или даже неделями на кластере современных графических процессоров.

1.1.3. Анализ EfficientNetV2

Для борьбы с этой проблемой в 2019 году было проведено исследование масштабирования моделей и балансирования между собой их глубины и ширины, а также разрешения входного изображения [16]. Авторы данного исследования предложили новый метод составного масштабирования (англ. *Compound scaling method*), который равномерно масштабировал данные параметры модели с описанными пропорциями между ними. В ходе экспериментов с данным методом масштабирования был автоматически

создан класс моделей EfficientNet, состоящий из восьми моделей, различающихся количеством параметров. Данные модели давали лучшую точность классификации при меньшем количестве параметров и времени обучения по сравнению с другими моделями своего времени. Архитектура базовой модели EfficientNet-B0 представлена на рисунке 7.

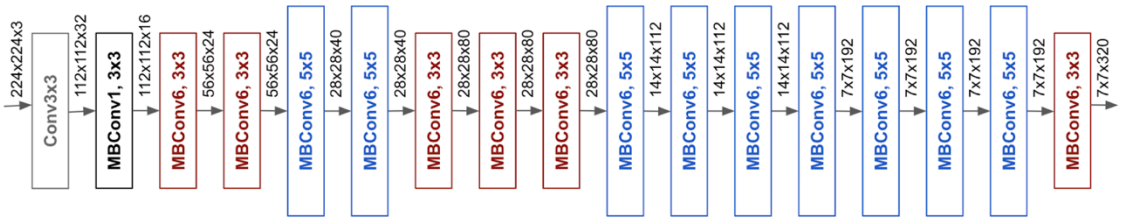

Рисунок 7. Архитектура модели EfficientNet-B0 [16]

В 2021 году авторы оригинальной статьи представили новый класс моделей – EfficientNetV2 [18], обладающих ещё более высокой скоростью обучения и меньшим количеством обучаемых параметров. График, демонстрирующий точность классификации моделями данного класса, а также время обучения по сравнению с другими современными моделями представлен на рисунке 8.

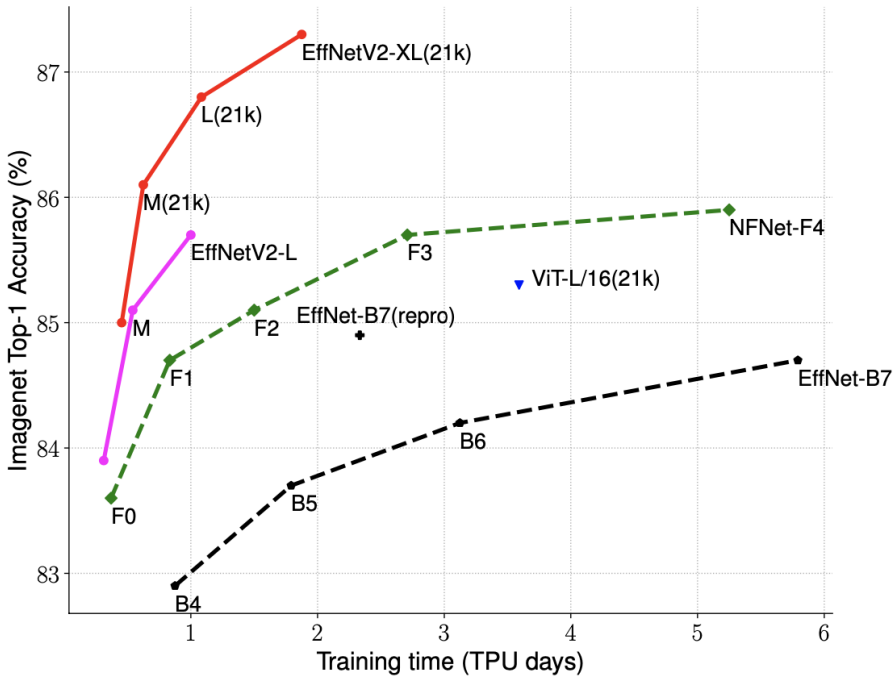

Рисунок 8. Сравнение точности классификации и скорости обучения моделей [18]

Главной отличительной особенностью второго поколения моделей от первого стало применение методики прогрессивного обучения (англ.

Progressive Learning), позволившей значительно сократить время обучения, при этом не ухудшив точность классификации. Суть данной методики заключается в динамическом увеличении разрешения изображений с каждой эпохой обучения, а также подстраивания параметров регуляризации под это разрешение. Регуляризация представляет собой набор различных методов, применяемых для затруднения обучения ИНС, тем самым помогающих бороться с эффектом переобучения. При обучении моделей данного семейства использовались следующие методы регуляризации:

- Отсев (англ. *Dropout*) – суть которого заключается в отбрасывании входных данных случайных нейронов в ходе обучения сети.
- Аугментация данных (англ. *Data Augmentation*) – расширение обучающего множества за счёт включения в него немного изменённых оригинальных изображений.
- Смешивание (англ. *Mixup*) – создание новых изображений на основе смешивания изображений обучающего множества, а также создание соответствующих им меток класса.

Всего в данный класс входят 8 моделей - EfficientNetV2-B0, EfficientNetV2-B1, EfficientNetV2-B2, EfficientNetV2-B3, EfficientNetV2-S, EfficientNetV2-M, EfficientNetV2-L и EfficientNetV2-XL, которые отличаются своими размерами. Так, например в модели EfficientNetV2-B0 чуть более семи миллионов обучаемых параметров, в то время как в EfficientNetV2-L их количество переваливает за сто восемнадцать миллионов.

Для достижения поставленной цели работы оптимальным выбором для выполнения практической части будет модель EfficientNetV2-B0. Этот выбор можно объяснить сравнительно небольшими размерами модели, а также тем фактом, что она уже обучена на наборе данных ImageNet и имеет на нём высокую точность. Останется только доучить модель на своём наборе данных, что не займёт много времени в силу её малого количества параметров.

1.2. Обзор методов байесовской статистики

Байесовская статистика является областью статистики, основанной на байесовской интерпретации вероятности, когда вероятность отражает степень уверенности в наступлении события. Ключевой особенностью байесовской статистики является то, что в степень уверенности можно вложить какие-то априорные знания о событии, основанные, например, на прошлых экспериментах или на каких-либо личных убеждениях исследователя о нём, а затем обновлять степень уверенности при появлении новых данных.

На протяжении большей части прошлого века байесовские методы почти не использовались в виду большого требуемого объема вычислений. Однако в 21 веке, с ростом мощности компьютеров и появлением новых алгоритмов, таких как марковские цепи Монте-Карло, байесовские методы начинают использоваться всё чаще.

1.2.1. Байесовский подход в машинном обучении

Байесовский подход в машинном обучении заключается в интерпретировании всех неизвестных величин (параметров модели) как случайных. Распределение параметров задаётся совместной плотностью их распределения. Благодаря тому, что значения параметров оцениваются не точечно, как в частотном подходе, а распределением, на выходе модели также получается распределение, что позволяет оценить неопределённость модели в сделанном предсказании.

Запишем формулу Байеса в терминах задачи моделирования данных:

$$p(\theta|D) = \frac{p(D|\theta)\cdot p(\theta)}{p(D)} = \frac{p(D|\theta)\cdot p(\theta)}{\int_\theta p(D|\theta)\cdot p(\theta)\partial\theta},$$

где θ – параметры модели;

$p(\theta)$ – априорная плотность распределения (англ. *prior*);

D – наблюдаемые данные;

$p(D)$ – вероятность наблюдать эти данные (англ. *evidencde*);

$p(D|\theta)$ – правдоподобие (англ. *likehood*);

$p(\theta|D)$ – апостериорная плотность распределения (англ. *posterior*);

После наблюдения новых данных, используя формулу Байеса, можно получить апостериорное распределение параметров, тем самым улучшив качество предсказания. Однако, интеграл, находящийся в знаменателе формулы Байеса, зачастую является неберущимся. Для решения этой проблемы используются марковские цепи Монте-Карло – класс алгоритмов для выборки из некоторого моделируемого распределения вероятностей. После построения марковской цепи, моделирующей апостериорную плотность распределения, можно будет получить выборку из апостериорной плотности вероятности, запоминая состояния цепи. Для построения цепей используются различные алгоритмы, одним из наиболее эффективных является NUTS (No U-Turn Sampler) [19]. Марковской цепи Монте-Карло нужно какое-то время, чтобы сойтись к заданному стационарному распределению – необходимому апостериорному распределению, поэтому на практике обычно откидывают первые полученные значения. Также на практике, для более достоверных результатов моделирования, обычно используют несколько цепей.

1.2.2. Байесовская логистическая регрессия

Одной из самых простых байесовских статистических моделей является логистическая регрессия [20]. Помимо своей простоты, данная модель выделяется тем, что она по сути повторяет собой последний слой СНС. Байесовская логистическая регрессия используется для бинарной классификации, обозначим метки классов как 0 и 1. Классификация осуществляется по следующей формуле:

$$p = \frac{1}{1+e^{-(\beta_0+\sum_{i=1}^{n}\beta_i \cdot x_i)}},$$

где $\beta_i,\ i = \overline{0,n}$ – коэффициенты регрессии;

$x_i,\ i = \overline{1,n}$ – значения наблюдаемых признаков;

n – количество наблюдаемых признаков;

p –вероятности принадлежности наблюдаемого объекта к 1 классу;

При $p \geq 0.5$ наблюдаемый объект относится к 1 классу, иначе – к классу с номером 0.

Построение модели начинается с задания априорных распределений плотностей коэффициентов регрессии. При отсутствии априорных знаний о их распределении, имеет смысл использовать нормальное или равномерное распределение. После чего для выходного параметра задаётся распределение Бернулли, которое в качестве вероятности принимает значение p. Далее запускаются марковские цепи Монте-Карло, которые моделируют значения коэффициентов регрессии, после чего моделируют значения выходного параметра модели. В качестве наблюдаемых данных используются примеры из обучающего множества.

Для решения нашей задачи имеет смысл использовать именно байесовскую логистическую регрессию, поскольку нам достаточно отделить рассматриваемый редких класс от остальных и решать задачу бинарной классификации. Помимо этого, обучение данной модели является относительно эффективным с точки зрения вычислительных затрат.

Глава 2. Реализация описанных моделей для классификации

Для выполнения практической части работы был выбран язык программирования Python. Выбор пал именно на этот язык программирования ввиду наличия большого количества библиотек для работы с ИНС и байесовскими методами. В качестве среды разработки (IDE) был выбран Jupyter Notebook, позволяющий работать с Python в интерактивном режиме.

2.1. Описание выбранного набора данных

В качестве набора данных для выполнения практической части работы был выбран набор изображений больных и здоровых листьев растений. Всего в данном наборе данных представлено 1641 изображение и 11 классов:

- Apple leaf miner (29 изображений)
- Healthy apple leaves (481 изображение)
- Healthy begonia leaves (162 изображения)
- Healthy coleus leaves (41 изображение)
- Grape leafhopper (75 изображений)
- Healthy grape leaves (210 изображений)
- Healthy Iresine leaves (39 изображений)
- Healthy kiwi leaves (31 изображение)
- Laurel leaf weevils (56 изображений)
- Laurel leaf thrips (37 изображений)
- Healhty violet leaves (490 изображений)

Как можно заметить, в данном наборе данных содержатся редкие классы, поэтому есть основания полагать, что СНС может ошибиться при классификации изображений из тестирующего множества, принадлежавших этим классам.

2.2. Разбиение исходного набора данных

Перед началом обучения СНС и построением байесовской модели необходимо разделить изначальный набор данных на три множества:

- Обучающее – используется для обучения модели.

- Подтверждающее – используется для оценки качества работы модели во время обучения.
- Тестирующее – используется для тестирования модели после процесса обучения.

Для разделения исходных данных на множества был написан скрипт на языке программирования Python. Из каждого класса в обучающее множество попало около 70% случайно выбранных изображений, а в подтверждающее и тестирующее – по 15% соответственно. При невозможности разделить изображения поровну, предпочтение отдавалось тестирующему множеству. Таким образом, в обучающем множестве оказалось 1154 изображения, в подтверждающем – 245 изображений, а в тестирующем – 253 изображения соответственно.

2.3. Создание модели свёрточной нейронной сети

В качестве базовой архитектуры СНС, как и было решено в прошлой главе, была выбрана EfficientNetV2B0. Данная модель уже реализована в библиотеке Keras [21], поэтому нет необходимости реализовывать и обучать модель самостоятельно. Однако, поскольку эта модель предобучена на наборе данных ImageNet, то есть решает задачу классификации на 1000 классов, её архитектуру придётся доработать под нашу задачу.

Для этого заменим последний полносвязный слой двумя новыми. Первый слой будет состоять из 22 нейронов, а в качестве функции активации выступит ReLU, второй же слой будет использоваться для классификации на 11 классов, поэтому к его выходам применится функция SoftMax. Решение добавить промежуточный полносвязный слой объясняется необходимостью снизить количество выходных значений перед последним полносвязным слоем сети. Эта необходимость возникает в связи с тем, что в дальнейшем его выходы будут использоваться для обучения и тестирования модели байесовской логистической регрессии и обучение при слишком большом количестве параметров будет крайне неэффективно. Изначальное количество нейронов на этом слое было взять как удвоенное количество нейронов

последнего слоя. Далее, в ходе экспериментов оказалось, что это количество оптимально по времени обучения и качества классификации обеих моделей.

Поскольку первые блоки СНС обнаруживают на входном изображении довольно низкоуровневые признаки, то с целью оптимизации времени обучения «заморозим» веса первых пяти блоков полученной модели. «Замороженные» веса фиксируются и не будут изменяться в ходе дальнейшего обучения модели. Теперь, в ходе обучения будут изменяться параметры только последнего – шестого блока, а также добавленных полносвязных слоёв. В итоге получили модель с чуть более четырёх с половинной миллионами обучаемых параметров.

2.4. Обучение и тестирование свёрточной нейронной сети

Теперь проведём обучение получившейся модели. Поскольку модель классифицирует исходные данные на 11 классов, в качестве функции потерь при обучении была выбрана перекрёстная энтропия. Обучение длилось 30 эпох на процессоре i9-9980HK, одна эпоха обучения занимала от 128 до 154 секунд. Значения параметров модели сохранялись после каждой эпохи.

Графики зависимости точности модели на обучающем и подтверждающем множествах, а также значения функций потерь представлены на рисунках 9 -10 .

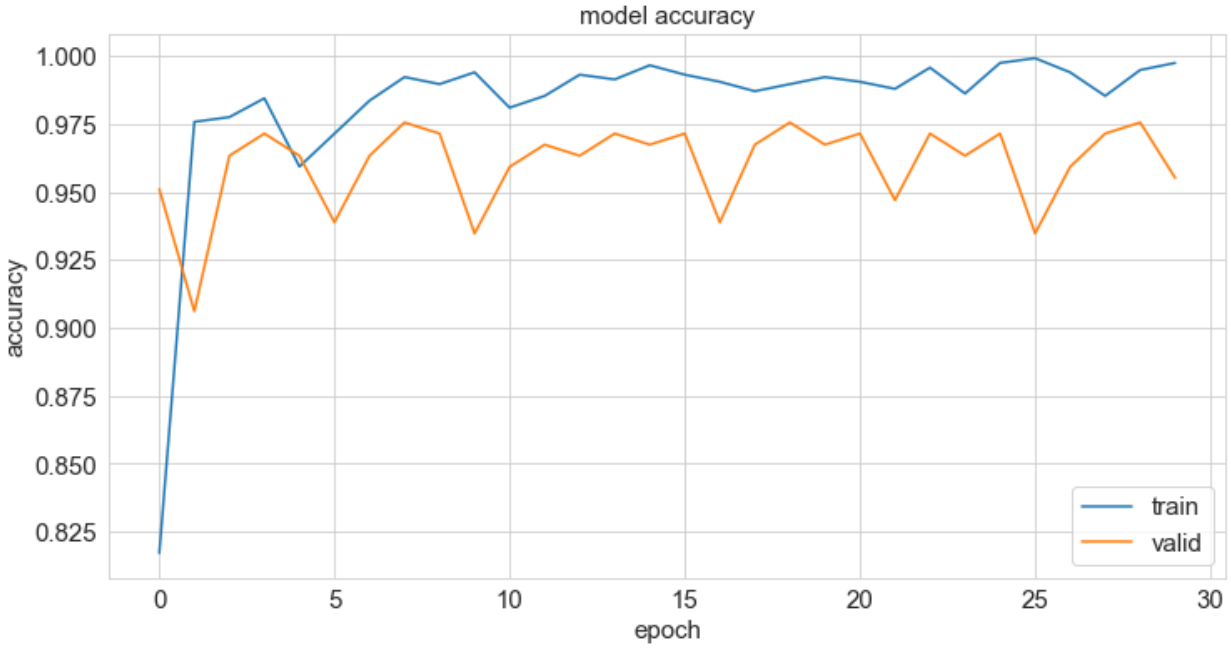

Рисунок 9 – точность при обучении модели

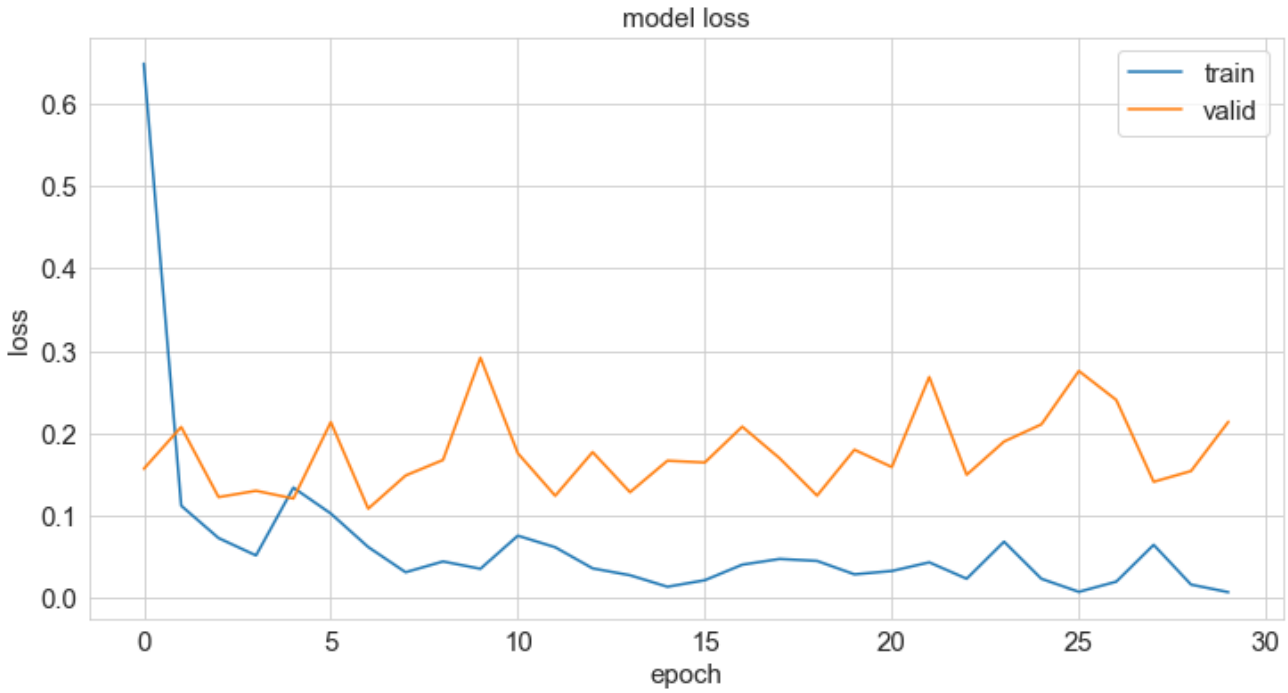

Рисунок 10 – значение функции потери при обучении модели

Как можно заметить из рисунков, модели потребовалось всего несколько эпох для того, чтобы обучиться на новых данных, а колебания её точности и значений функции потерь объясняются стохастической природой алгоритма обучения.

Максимальная точность на подтверждающем множестве составила 97.55%. Модель продемонстрировала такую точность после 8, 19 и 29 эпох обучения. Для дальнейшего исследования была выбрана модель после 19 эпох обучения, поскольку значение функции потерь на подтверждающем множестве у неё оказалось минимальным.

Далее модель была использована для классификации изображений из тестирующего множества. Точность классификации составила 97.23%. Для обнаружения ошибок классификации была построена матрица неточностей (англ. *Confusion Matrix*), представленная на рисунке 11 .

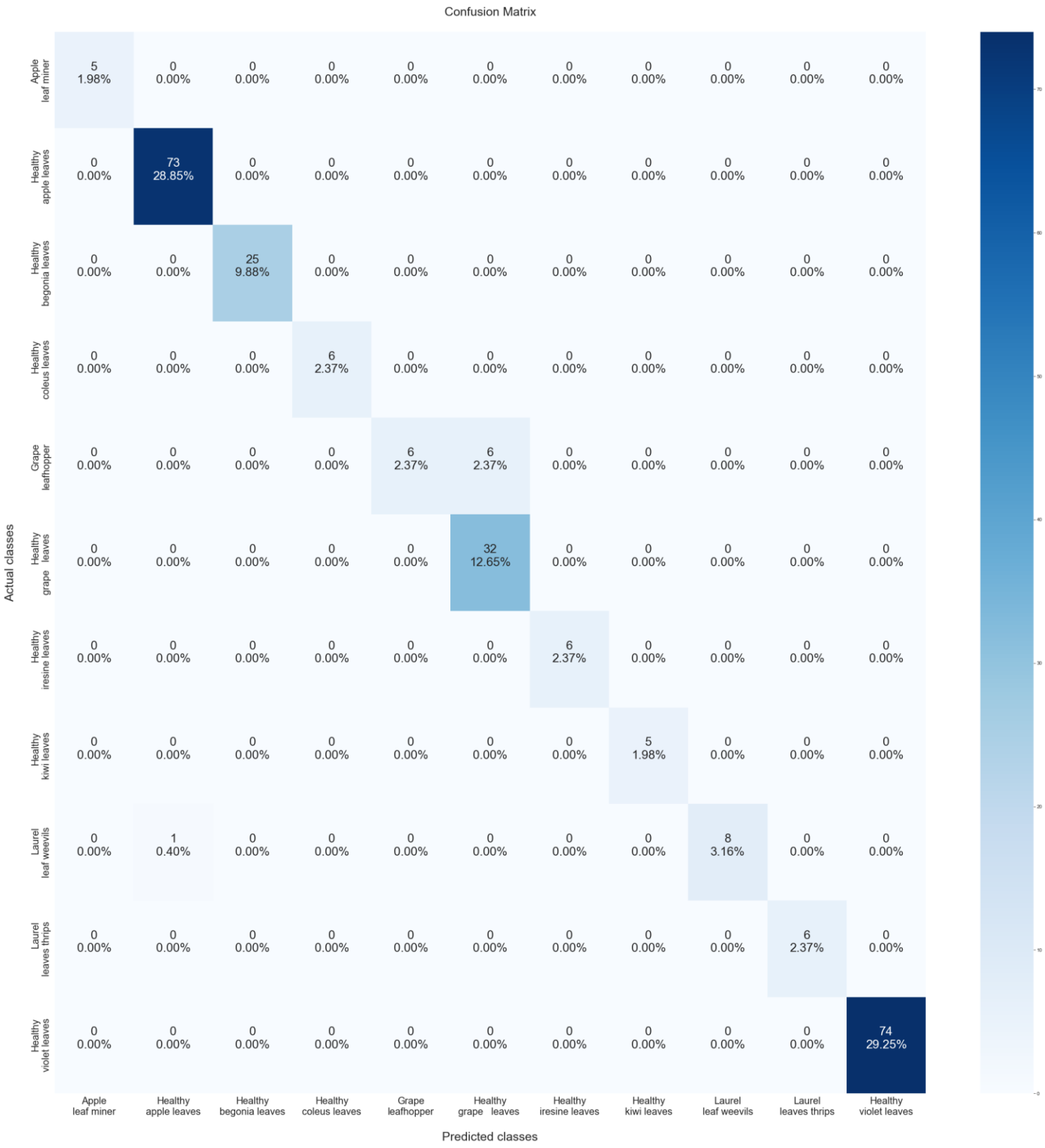

Рисунок 11 – матрица неточностей

Видно, что СНС абсолютно верно классифицировала девять классов из одиннадцати, в одном классе ошиблась в одном примере из девяти, однако в одном из классов точность классификации составила всего 50% (6 верно классифицированных примеров из 12).

2.5. Реализация байесовской логистической регрессии

В качестве байесовской статистической модели для классификации была выбрана логистическая регрессия.

2.5.1. Подготовка входных данных для модели

Для дальнейшего обучения и тестирования байесовской модели необходимо подготовить входные данные. Входными данными для байесовской логистической модели будут являться значения на выходе предпоследнего слоя СНС, состоящего из 22 нейронов. Также необходимо хранить метки классов, к которым принадлежат изображения. Помимо этого, для сравнения результатов СНС и байесовской модели, будем хранить векторы предсказаний, выданных СНС на тестирующем множества. Таким образом, формируем две таблицы входных данных для байесовской модели. В первой таблице, соответствующей тренировочному множеству, будет 23 столбца (22 значения признаков и метки классов) и 1154 строки. Во второй таблице, в свою очередь, будет 34 столбца (22 значения признаков, метки классов, а также 11 значений – вероятностей принадлежности к классам) и 253 строки. Для дальнейшего манипулирования этими данными удобно воспользоваться библиотекой Pandas [22], предоставляющей структуру данных DataFrame, соответствующую двумерной таблице.

2.5.2. Создание модели

Теперь перейдём непосредственно к реализации модели. В тестирующем наборе данных было 2 класса, примеры которых были классифицированы неверно. Поэтому будем строить 4 модели байесовской логистической регрессии – по модели для истинных классов, а также по модели для классов, к которым были отнесены данные примеры (классы Healthy apple leaves, Grape leafhopper, Healthy grape leaves, Laurel leaf weevils) Для создания моделей воспользуемся библиотекой PyMC3 [23], созданной специально для построения байесовских статистических моделей. Для начала необходимо сформировать векторы меток классов для каждой модели, поставив в соответствие интересующему нас классу 1, а всем остальным – 0.

Далее описывается процесс создания одной модели байесовской логистической регрессии, но он идентичен и для остальных, за исключением значений вектора меток классов. Сначала нужно задать априорные распределения коэффициентов регрессии. В ходе экспериментов было выявлено, что оптимальным вариантом для всех моделей будет использовать нормальное распределение с нулевым математическим ожиданием и среднеквадратичном отклонение 1 у первого коэффициента регрессии и 0.125 у остальных. Далее вводится функция сигмоиды, принимающая в качестве аргумента сумму первого коэффициента регрессии и линейной комбинации остальных коэффициентов и значений признаков наблюдаемого объекта. После этого вводится выходной параметр, распределённый по закону Бернулли и принимающий в качестве наблюдаемых данных, вектор меток классов. Далее запускаются 3 марковских цепи Монте-Карло, моделирующих по 11000 значений из апостериорных распределений заданных параметров модели, после чего откидывающих первую 1000 значений.

Таким образом, получили по 30000 значений из апостериорных распределений для каждого параметра модели.

Повторив этот процесс четыре раза, получим модели для всех интересующих нас классов. Теперь, используя эти модели, можем построить апостериорные плотности распределения вероятностей принадлежности к классам и сравнивать полученные результаты с прогнозами СНС.

Глава 3. Анализ результатов классификации

Перейдём к сравнению результатов, полученных свёрточной нейронной сетью и байесовской логистической регрессией. Имеет смысл начать с класса Grape leafhopper, поскольку точность классификации данного класса на тестирующем множестве составила всего 50%. Стоит отметить, что все неверно классифицированные примеры были распознаны как Healthy grape leaves, то есть СНС принимала больные виноградные листья за здоровые.

3.1. Анализ классификации класса Grape leafhopper

Начнём анализ с построения графиков байесовских апостериорных распределений вероятности принадлежности к классу Grape leafhopper и прогнозов СНС на неверно классифицированных примерах обучающего множества. Данные графики представлены на рисунке 12 .

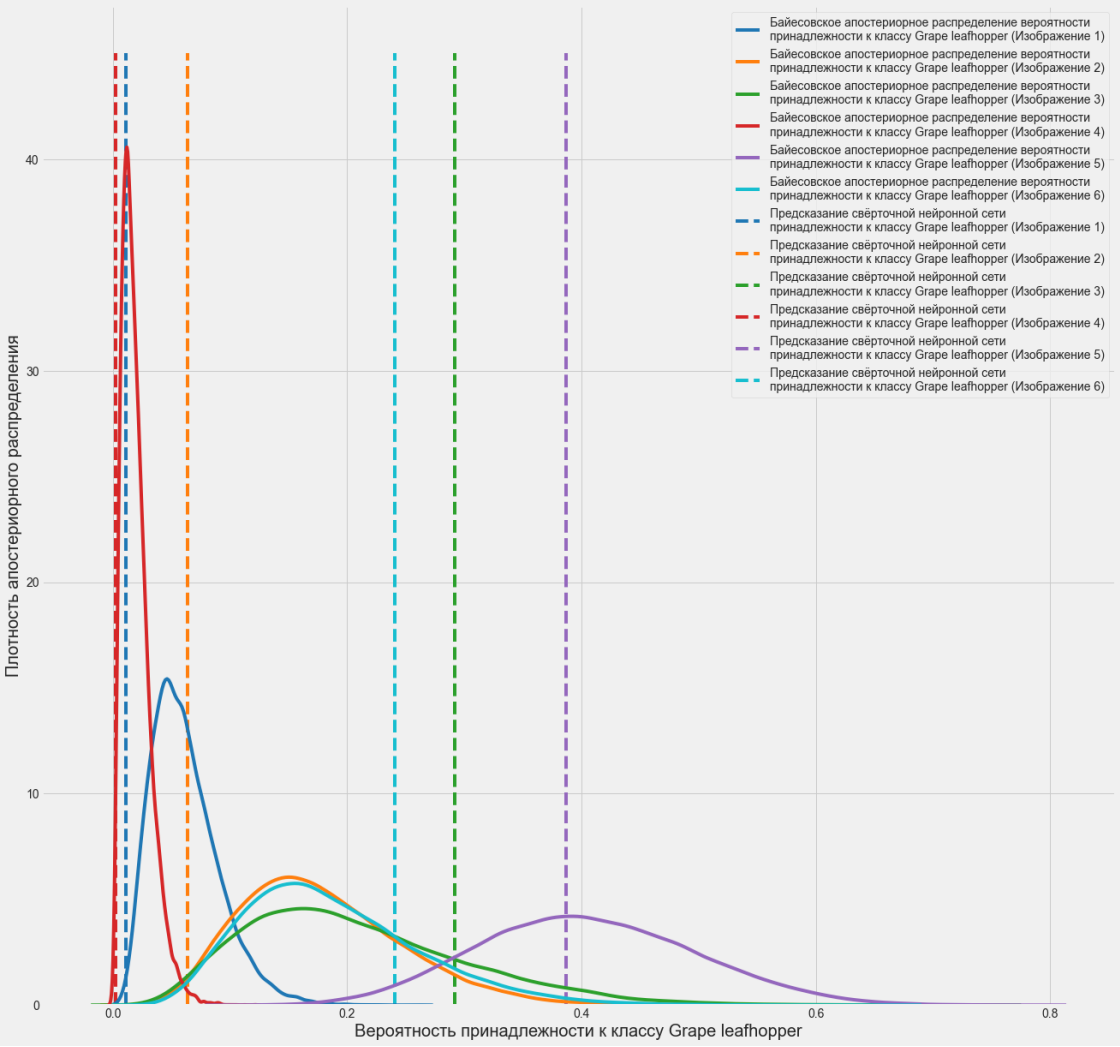

Рисунок 12 – сравнение предсказаний на неверно классифицированных примерах обучающего множества

Как видно на рисунке, апостериорные распределения вероятностей для большинства примеров лежат в довольно большом диапазоне, что свидетельствует о высокой неопределённости.

Рассмотрим несколько неверно классифицированных примеров более детально. Начнём сравнение с изображения 1 из тестирующего множества. Данное изображение представлено на рисунке 13.

Рисунок 13 – неверно классифицированное изображение 1

Вероятность принадлежности данного изображению к своему классу, выданная СНС, равна 1.1%, в то время как к классу здоровых листьев – примерно 98.77%. Если приглядеться, то на данном листе можно заметить повреждения, наиболее выраженные в левой и правой частях листа. Однако, неподготовленный человек с большой долей вероятности, как и СНС, может ошибиться и принять данный лист за здоровый.

Теперь построим график, на который нанесём байесовские апостериорные распределения вероятностей принадлежности к классам, а также результаты, выданные СНС. Данный график продемонстрирован на рисунке 14 .

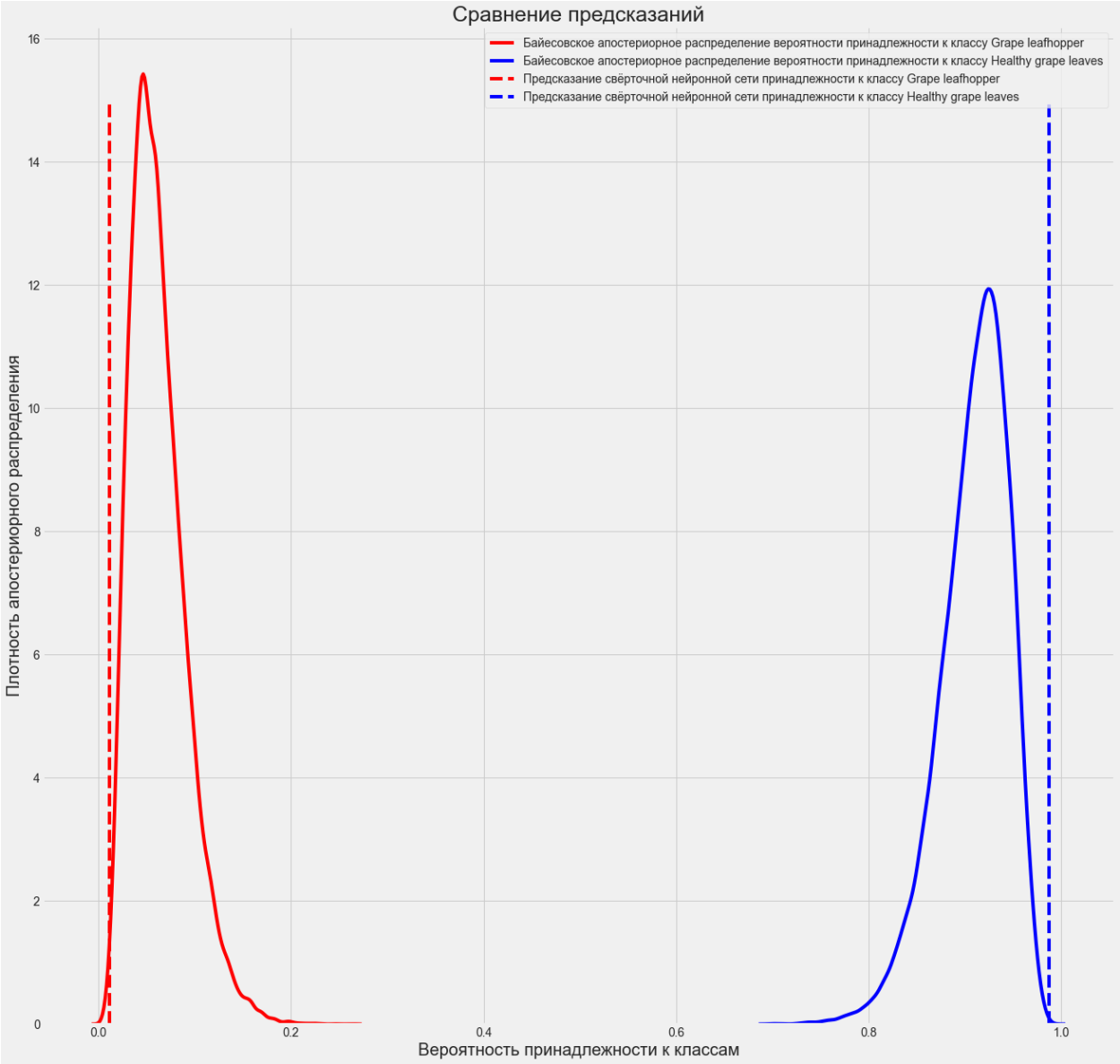

Рисунок 14 – сравнение предсказаний на первом неверно классифицированном примере

Как можно заметить на рисунке, средние значения апостериорных распределений вероятности принадлежности к классам находятся ближе друг к другу, нежели предсказания СНС. Так, среднее значение вероятности принадлежности к классу Grape leafhopper составляет около 6.18%, в то время как средняя вероятность, что данный лист здоров – 90.92%. Также, в отличие от результата СНС у нас есть не точечное предсказание вероятности, а

диапазон, в котором сосредоточена вероятность правильного предсказания – от 0 до примерно 20%.

Теперь рассмотрим неверно классифицированное изображение 3 из тестирующего множества. Данное изображение представлено на рисунке 15 .

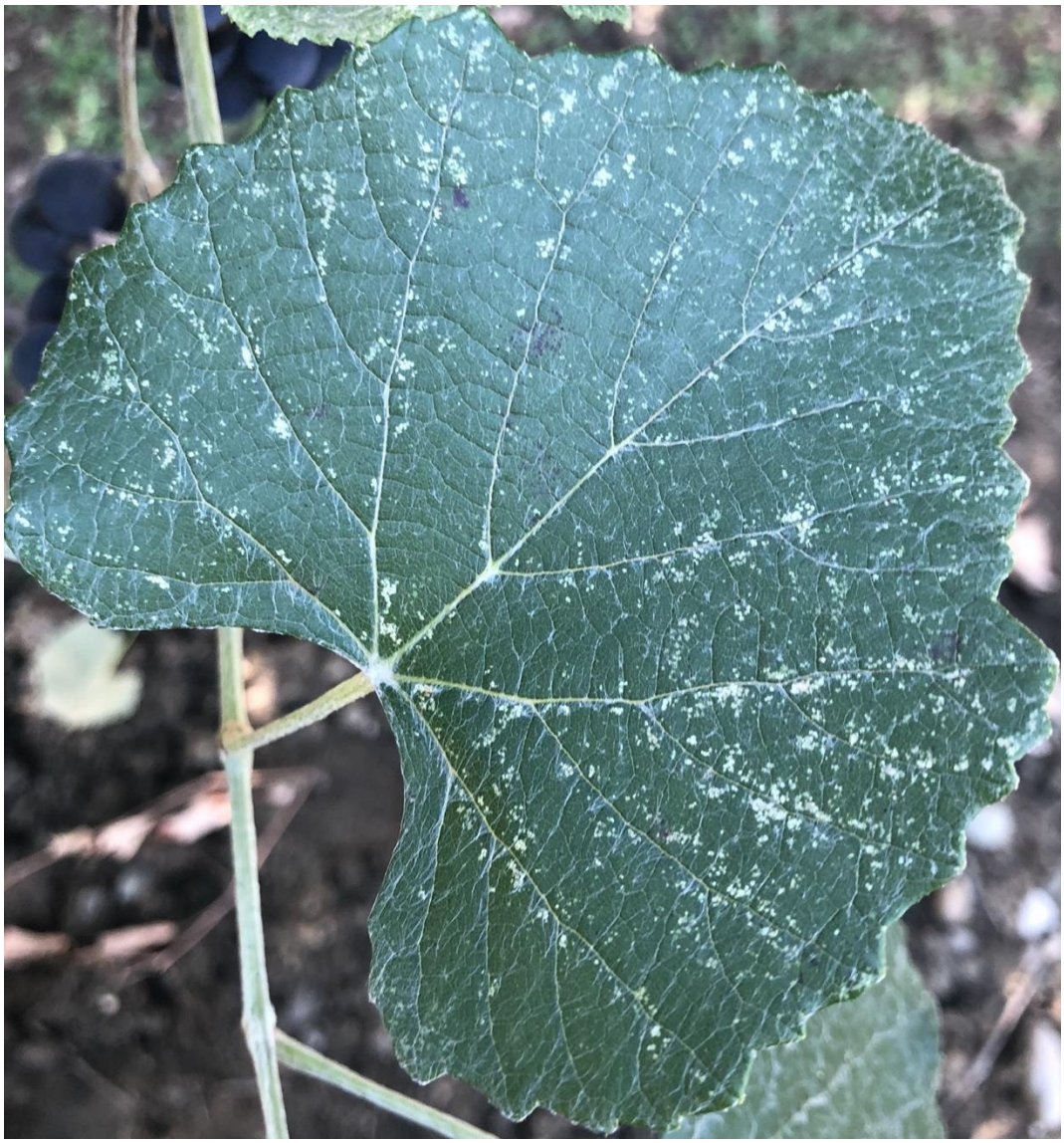

Рисунок 15 – неверно классифицированное изображение 3

На данном примере очевидно наличие повреждений на листе, однако СНС отнесла его к здоровым с вероятностью 93.09%. Вероятность наличия на листе повреждений, в свою очередь, составила всего 6.42%.

Для сравнения результатов СНС и байесовской логистической регрессии построим график, аналогичный прошлому примеру. Данный график изображён на рисунке 16 .

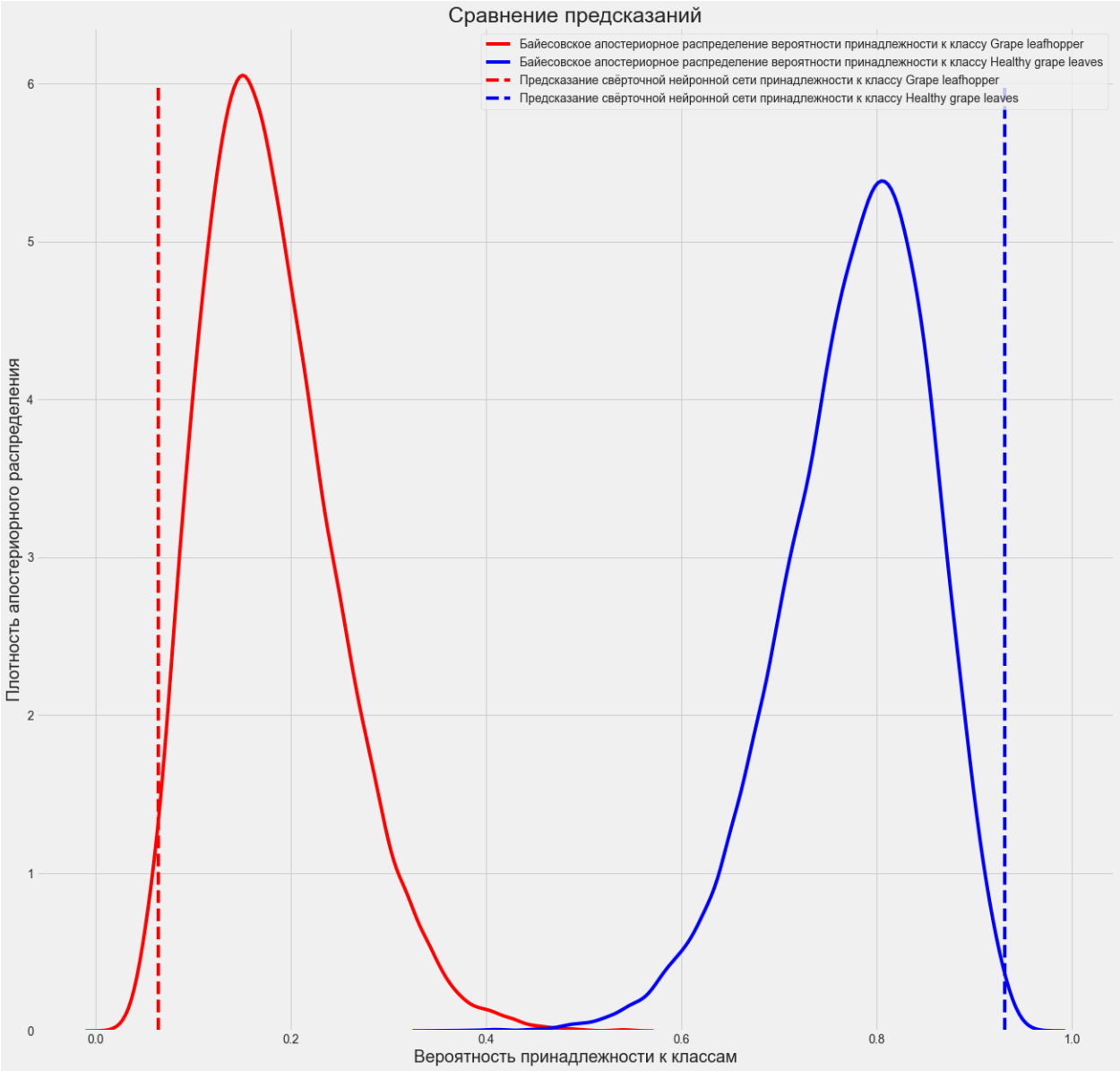

Рисунок 16 – сравнение предсказаний на третьем неверно классифицированном примере

Как видно из рисунка, предсказание байесовской модели значительно отличается от прогноза СНС. Так, среднее значение вероятности правильного предсказания составляет около 17.82%, в то время как средняя вероятность ошибиться – 77.75%. Также на рисунке видно, в насколько большом диапазоне лежат значения апостериорных распределений вероятностей, что говорит о том, что модель очень не уверена в предсказании.

Теперь построим графики байесовских апостериорных распределений вероятности принадлежности к классу Grape leafhopper и прогнозов СНС на верно классифицированных примерах обучающего множества. Данные графики представлены на рисунке 17.

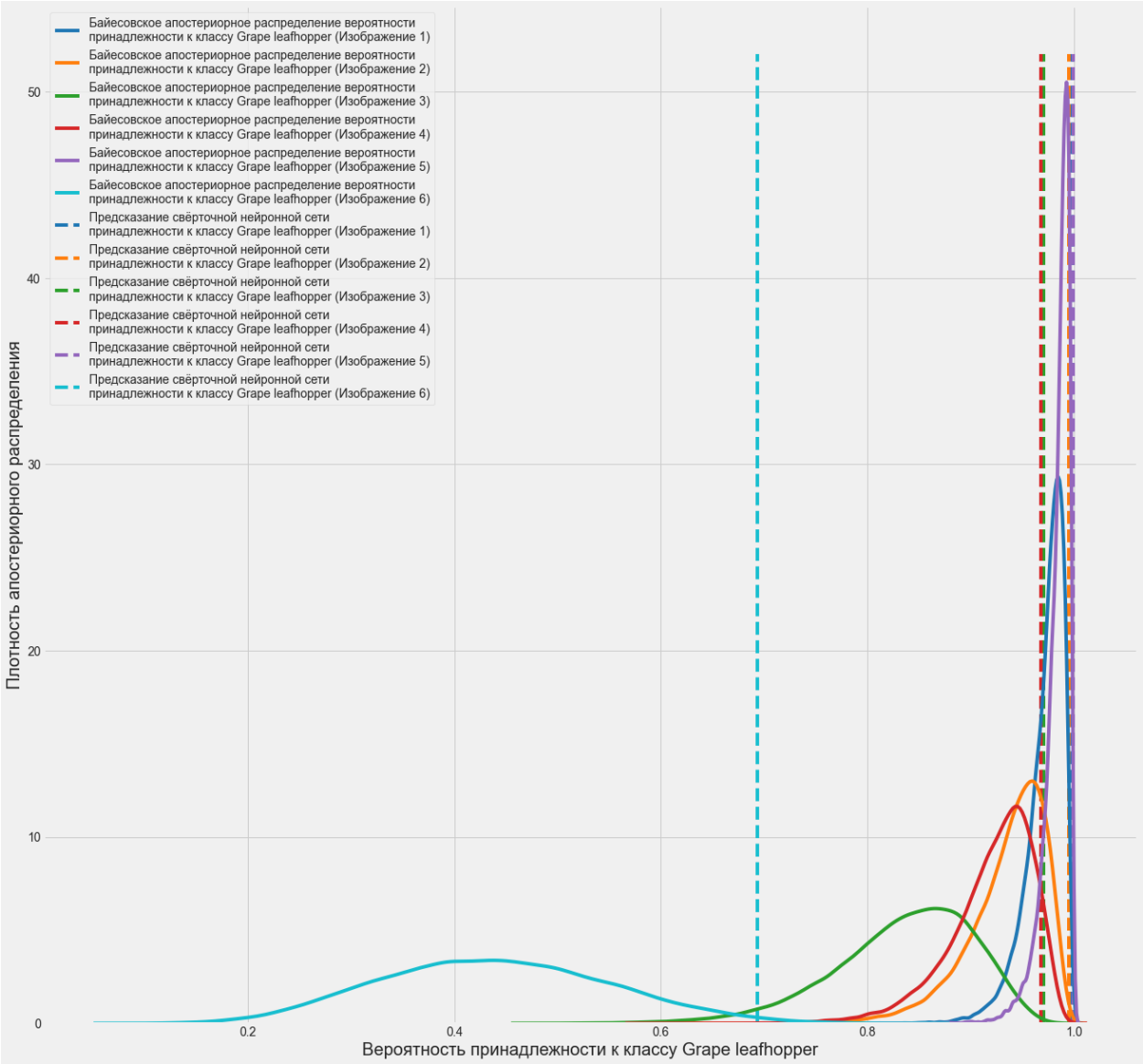

Рисунок 17 – сравнение предсказаний на верно классифицированных примерах обучающего множества

Как и в случае с неверно классифицированными изображениями, видно, что апостериорные распределения вероятностей, для некоторых изображений, лежат в довольно большом диапазоне

Наиболее интересным примером для более детального анализа здесь выглядит шестое изображение. Данное изображение представлено на рисунке 18.

Рисунок 18 – пример верно классифицированного изображения

Неуверенность модели в классификации этого изображения можно объяснить почти что полным отсутствием характерных чёрных пятен на листе. Свёрточная нейронная сеть классифицирует этот лист как повреждённый с вероятностью 69.32%, вероятность того, что лист здоров – 7.68%.

Теперь построим график байесовской апостериорного распределения вероятности классификации данного изображения к классам Grape leafhopper и Healthy grape leaves, а также нанесём на него результаты СНС. Данный график представлен на рисунке 19.

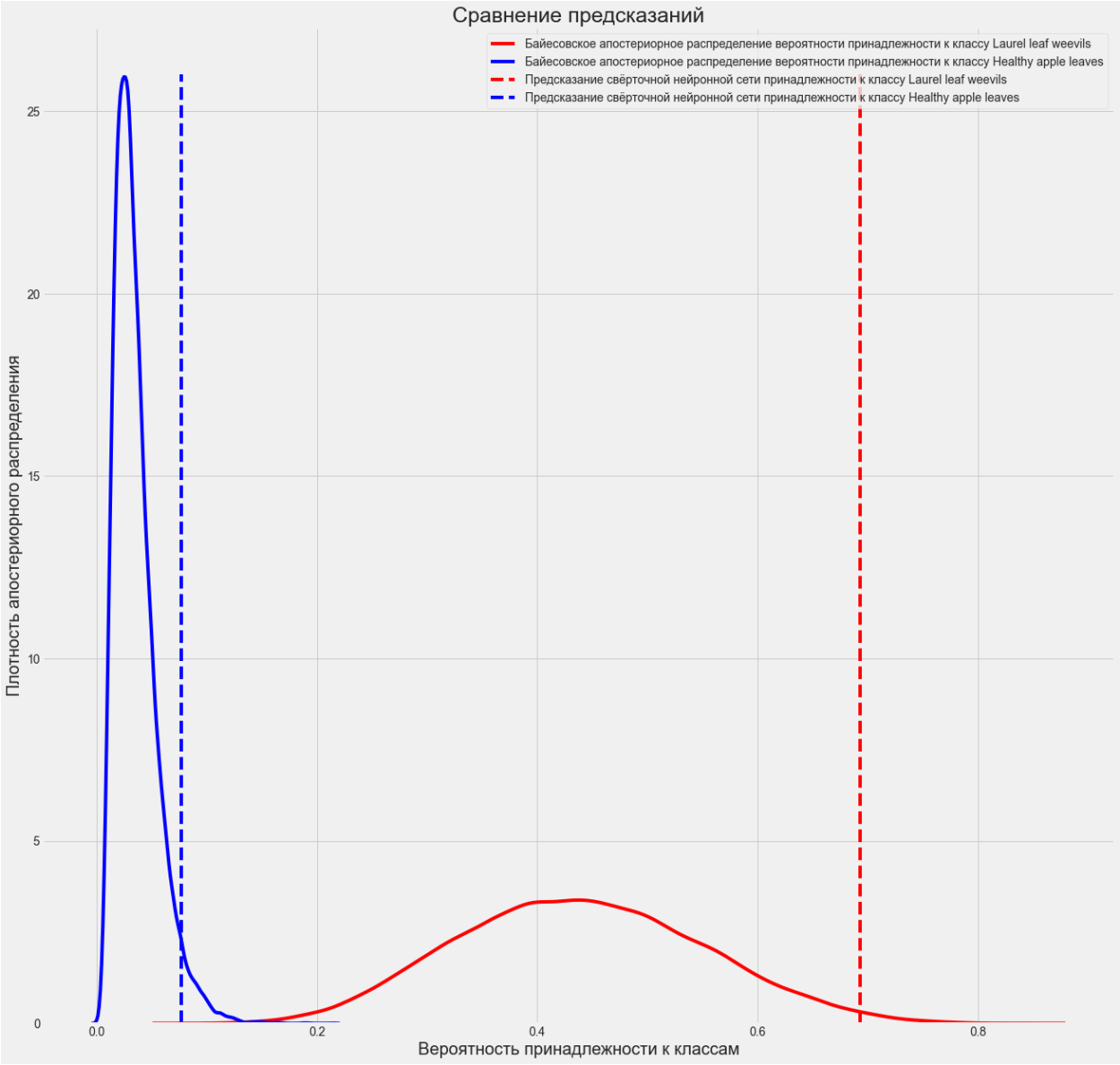

Рисунок 19 – сравнение предсказаний на шестом верно классифицированном примере

Как можно заметить, вероятность принадлежности данного изображения к классу Grape leafhopper расположена примерно от 0 до 90%, со средним значением 43.92%. Это говорит о большой неуверенности модели в предсказании. Среднее значение вероятности того, что лист здоров составляет около 7.68%, что совпадает с результатом СНС.

3.2. Анализ классификации класса Laurel leaf weevils

Помимо ошибок при классификации листьев винограда, СНС также один раз ошиблась и на другом классе, приняв повреждённый лавровый лист за здоровый лист яблока. Данное изображение представлено на рисунке 20.

Рисунок 20 – Неверно классифицированный пример класса Laurel leaf weevils

На этом изображении видны только пятна на листе, что сильно отличает его от типичных примеров изображений своего класса. Пример типичного представителя класса Laurel leaf weevils представлен на рисунке 21.

Рисунок 21 – Типичный пример класса Laurel leaf weevils

Этим объясняется тот факт, что СНС с вероятностью в 70.55% отнесла данное изображение к Healthy apple leaves, и только с вероятностью в 12.19% к классу Laurel leaf weevils.

Для сравнения прогнозов СНС и байесовской логистической регрессии построим график апостериорного распределения вероятности классификации данного изображения к классам Laurel leaf weevils и Healthy apple leaves, а также нанесём на него результаты работы СНС. Данный график представлен на рисунке 22 .

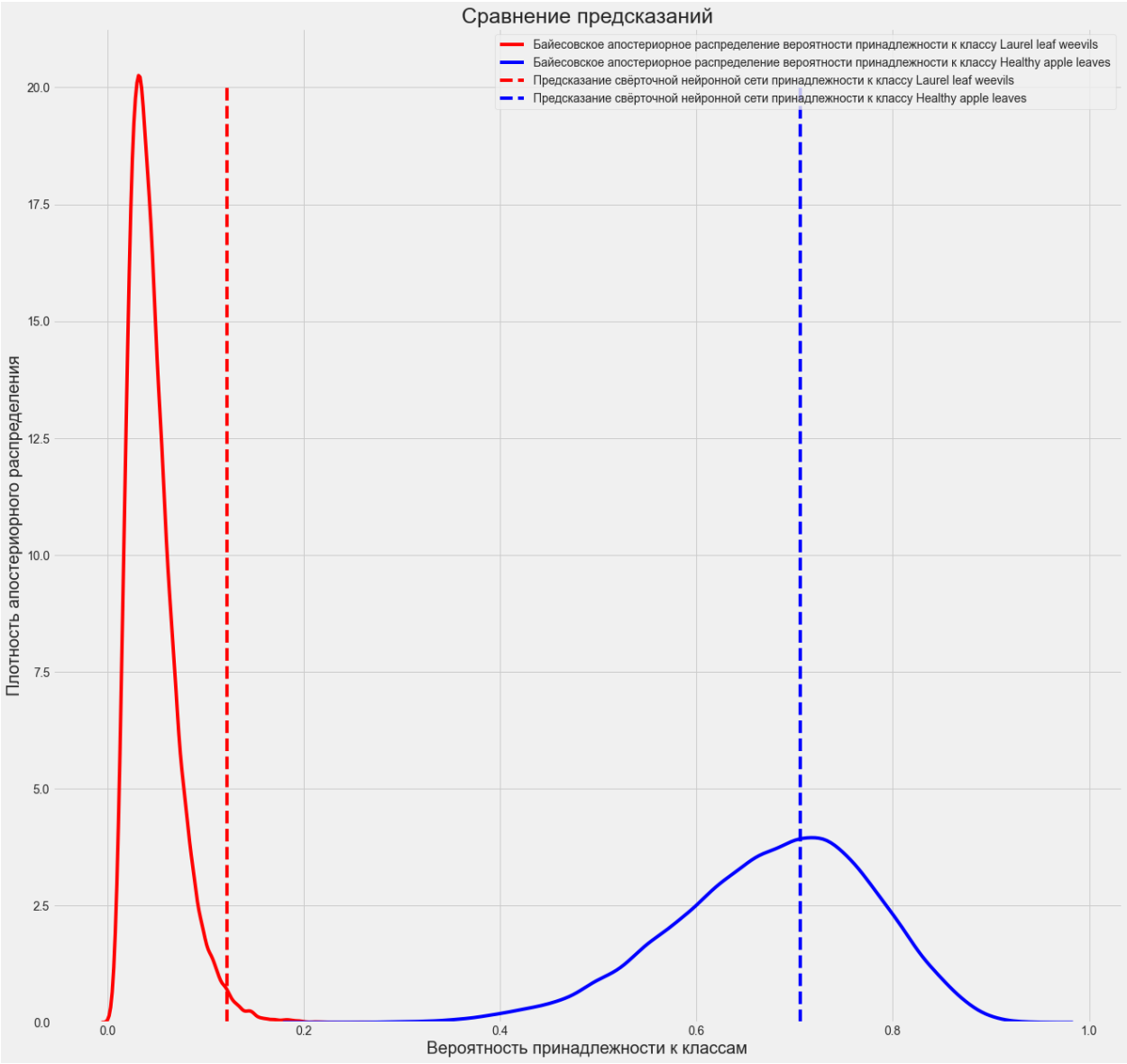

Рисунок 22 – сравнение предсказаний на примере класса Laurel leaf weevils

Средняя вероятность принадлежности данного изображения к классу Laurel leaf weevils составляет 4.71%, что почти втрое ниже, чем результат СНС, а средняя вероятность классификации этого изображения как Healthy apple

leaves составляет 67.75%, что незначительно ниже прогноза свёрточной нейронной сети. Несмотря на это, можно заметить, что вероятность принадлежности данного изображения к классу Healthy apple leaves довольно сильно рассредоточена, что говорит о высоком уровне неопределённости, получив информацию о котором, можно будет принять соответствующие действия.

3.3. Выводы

По результатам проведённых экспериментов можно сделать вывод о том, что СНС, ввиду использования функции SoftMax довольно часто либо слишком занижает настоящую вероятность, либо слишком завышает её. Также СНС не сообщает никакой информации о степени неопределённости при прогнозировании. Использование байесовских методов, в большинстве случаев, даёт более точное представление о вероятности принадлежности к классу, а также показывает неопределённость в предсказании.

Заключение

В ходе выполнения данной работы была создана свёрточная нейронная сеть, а также байесовская логистическая регрессия. Было проведено сравнение их результатов при классификации редких классов изображений для проверки выдвинутой гипотезы.

Для достижения поставленной цели был проведён анализ современных моделей СНС, в качестве оптимальной модели по соотношению качества классификации и скорости обучения для выполнения практической части работы была выбрана архитектура EfficientNetV2B0.

Данная модель была обучена и протестирована на наборе данных о болезнях листьев растений. В данном наборе данных содержится 11 классов, среди которых есть редкие. Точность классификации на тестирующем множестве составила 97.23%.

Далее был проведён небольшой обзор байесовской статистики, а также выбор оптимальной модели для выполнения практической части работы. В качестве такой модели была выбрана байесовская логистическая регрессия. Были созданы модели для классификации редких классов изображений, при классификации которых СНС допустила ошибки.

Было проведено сравнение результатов классификации редких классов изображений. В ходе сравнения было выявлено, что свёрточная нейронная сеть склонна либо слишком сильно занижать вероятность принадлежности объекта к классу, либо слишком сильно завышать её. Байесовские методы классификации, в свою очередь, строят плотность распределения вероятности, что помогает оценить неопределённость модели в прогнозе. Данная оценка может быть полезна в областях, где уверенность в предсказании критично важна, например, в медицине, при диагностировании заболеваний. Таким образом, можно сделать вывод о принятии выдвинутой гипотезы.

Список использованных источников

1. IBM. What is computer vision? [Электронный ресурс]. URL: https://www.ibm.com/topics/computer-vision (дата обращения: 12.04.2022).
2. Google Lens [Электронный ресурс]. URL: https://lens.google/ (дата обращения: 14.04.2022).
3. Viso.ai. A Complete Guide to Image Classification in 2022 [Электронный ресурс]. URL: https://viso.ai/computer-vision/image-classification/ (дата обращения: 14.04.2022).
4. Hubel D.H., Wiesel T.N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex // The Journal of Physiology. 1962. Vol. 160, № 1. P. 106–154.
5. Hubel D.H., Wiesel T.N. Receptive fields and functional architecture of monkey striate cortex // The Journal of Physiology. 1968. Vol. 195, № 1. P. 215–243.
6. LeCun Y. et al. Backpropagation Applied to Handwritten Zip Code Recognition // Neural Computation. 1989. Vol. 1, № 4. P. 541–551.
7. MLearning.ai. LeNet and MNIST handwritten digit recognition [Электронный ресурс]. URL: https://medium.com/mlearning-ai/lenet-and-mnist-handwritten-digit-classification-354f5646c590 (дата обращения 15.04.2022).
8. Rawat W., Wang Z. Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review // Neural Computation. 2017. Vol. 29, № 9. P. 2352–2449.
9. Nwankpa C. et al. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. 2018.
10. Nair V., Hinton G.E. Rectified linear units improve restricted boltzmann machines // ICML'10: Proceedings of the 27th International Conference on International Conference on Machine Learning. 2010. P. 807-814.

11. Brilliant. Backpropagation [Электронный ресурс]. URL: https://brilliant.org/wiki/backpropagation/ (дата обращения: 15.04.2022).
12. Papers With Code. Image Classification on ImageNet [Электронный ресурс]. URL: https://paperswithcode.com/sota/image-classification-on-imagenet (дата обращения: 16.04.2022).
13. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks // Commun ACM. 2017. Vol. 60, № 6. P. 84–90.
14. Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014.
15. He K. et al. Deep Residual Learning for Image Recognition // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2016. P. 770–778.
16. Tan M., Le Q. v. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. 2019.
17. Ai.googleblog. EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling [Электронный ресурс]. URL: https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html (дата обращения: 20.04.2022).
18. Tan M., Le Q. v. EfficientNetV2: Smaller Models and Faster Training. 2021.
19. Hoffman M.D., Gelman A. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. 2011.
20. Towards Data Science. Introduction to Bayesian Logistic Regression [Электронный ресурс]. URL: https://towardsdatascience.com/introduction-to-bayesian-logistic-regression-7e39a0bae691 (дата обращения: 20.04.2022).
21. Keras [Электронный ресурс]. URL: https://keras.io/api/ (дата обращения: 20.04.2022).

22. Pandas [Электронный ресурс] URL: https://pandas.pydata.org/pandas-docs/stable/ (дата обращения 20.04.2022).
23. PyMC3 [Электронный ресурс] URL: https://docs.pymc.io/en/v3/ (дата обращения 20.04.2022).