

Программные системы и вычислительные методы

Правильная ссылка на статью:

Бондаренко В.А., Попов Д.И. Исследование и разработка алгоритмов к формированию эффективного ансамбля сверточных нейронных сетей для классификации изображений // Программные системы и вычислительные методы. 2024. № 1. DOI: 10.7256/2454-0714.2024.1.69919 EDN: WZDHQO URL: https://nbpublish.com/library_read_article.php?id=69919

Исследование и разработка алгоритмов к формированию эффективного ансамбля сверточных нейронных сетей для классификации изображений

Бондаренко Валерий Александрович

магистр, кафедра информационных технологий и математики, Сочинский государственный университет

354002, Россия, Краснодарский край, г. Сочи, ул. Верхняя Лысая Гора, 64

✉ valeriybbond@mail.ru



Попов Дмитрий Иванович

доктор технических наук

профессор, кафедра информационных технологий и математики, Сочинский государственный университет

354002, Россия, Краснодарский край, г. Сочи, ул. Политехническая, 7

✉ damity@mail.ru



[Статья из рубрики "Базы знаний, интеллектуальные системы, экспертные системы, системы поддержки принятия решений"](#)

DOI:

10.7256/2454-0714.2024.1.69919

EDN:

WZDHQO

Дата направления статьи в редакцию:

20-02-2024

Аннотация: Объектом исследования являются искусственные нейронные сети (ИНС) со сверточной архитектурой для классификации изображений. Предметом исследования является исследование и разработка алгоритмов построения ансамблей сверточных нейронных сетей (СНС) в условиях ограниченности обучающей выборки. Целью

исследования является выработка алгоритма к формированию эффективной модели на основе ансамбля сверточных СНС с применением методов усреднения результатов каждой модели, способную избежать переобучения в процессе повышения точности прогноза и обученную на небольшом объеме данных, меньше 10 тысяч примеров. В качестве базовой сети в составе ансамбля выработана эффективная архитектура СНС, которая показала хороший результат в качестве одиночной модели. Также в статье исследованы методы объединения результатов моделей ансамбля и даны рекомендации к формированию архитектуры СНС. В качестве методов исследования используется теория нейронных сетей, теория машинного обучения, искусственного интеллекта, использованы методы алгоритмизации и программирования моделей машинного обучения, использован сравнительный анализ моделей, построенных на разных алгоритмах с использованием классического ансамблирования с простым усреднением и объединения результатов базовых алгоритмов в условиях ограниченности выборки с учетом средневзвешенного усреднения. Областью применения полученного алгоритма и модели является медицинская диагностика в лечебных учреждениях, санаториях при проведении первичного диагностического приема, на примере исследовательской задачи модель обучена классификации дерматологических заболеваний по входным фотоснимкам. Новизна исследования заключается в разработке эффективного алгоритма и модели классификации изображений на основе ансамбля сверточных СНС, превосходящих точность прогноза базовых классификаторов, исследован процесс переобучения ансамбля классификаторов с глубокой архитектурой на малом объеме выборки из чего сделаны выводы по проектированию оптимальной архитектуры сети и выбору методов объединения результатов нескольких базовых классификаторов. По итогу исследования разработан алгоритм к формированию ансамбля СНС на основе эффективной базовой архитектуры и средневзвешенного усреднения результатов каждой модели для классификационной задачи по распознаванию изображений в условиях ограниченности выборки.

Ключевые слова:

искусственные нейронные сети, сверточная архитектура, ансамбли нейронных сетей, методы усреднения результатов, задача классификации, медицинская диагностика, ансамблирование нейронных сетей, средневзвешенное усреднение, предварительная обработка, взвешенное голосование

Введение

Для повышения точности прогноза и уменьшения ошибок можно использовать ансамбли моделей нейронных сетей [\[1\]](#). Ансамбль – это комбинация нескольких моделей, работающих параллельно или последовательно и создающих объединенное предсказание на основе результатов каждой отдельной модели. Ансамбли алгоритмов машинного обучения представляют собой мощную технику, которая позволяет повысить точность и надежность классификации данных. Основная идея ансамблирования заключается в объединении нескольких базовых классификаторов таким образом, чтобы их индивидуальные ошибки компенсировались, а общая производительность системы улучшалась [\[2\]](#). Существует несколько различных подходов к ансамблированию, каждый из которых имеет свои преимущества и недостатки. Как правило, ансамблированные классификаторы превосходят отдельные базовые классификаторы по точности и надежности. Ансамблирование алгоритмов машинного обучения находит широкое

применение в различных областях, таких как распознавание образов, обработка естественного языка, биоинформатика и финансовый анализ. Однако ансамблирование имеет некоторые недостатки, например, увеличение вычислительной сложности, потенциально более высокая склонность к переобучению, при недостаточной выборке, сложность выбора подходящего метода ансамблирования. В рамках исследования решается задача многоклассовой одновариантной классификации изображений, обучение модели относится к обучению с учителем и осуществляется на основе размеченного набора данных [3]. В ходе исследования мы сравним результаты нескольких моделей, первый классификатор построен на базе простого усреднения результатов от нескольких моделей ансамбля СНС с применением глубоких предварительно обученных сетей, второй классификатор будет разработан на основе одной эффективной архитектуры сети, на основе которой будет реализовано несколько моделей ансамбля, обученных на независимых выборках с применением средневзвешенного объединения результатов моделей.

Материалы и методы

Основной проблемой в использовании глубоких нейронных сетей на малом объеме данных является переобучение, это такое явление когда сеть очень хорошо прогнозирует на обучающей выборке и плохо на тестовой. Основными показателями переобучения сети является очень большая разница показателя качества на обучающей и валидационной выборке [3], также признаком переобучения является неизменность показателя качества от эпохи к эпохе рисунок 1.

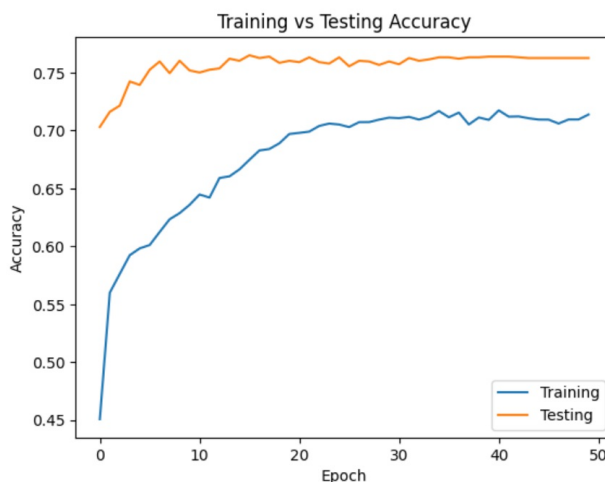


Рисунок 1. Стагнация показателя точности при обучении

Мы видим, что начиная с 30 эпохи обучения изменение показателя практически не происходит, в этом случае происходит малое изменение весовых коэффициентов, это значит, что мы находимся в области малых градиентов или достигли локального минимума [5]. Также причиной переобучения является неоднородность при распределении данных на тестовую и обучающую выборку, один класс получил нормальное соотношение отрицательного к положительному результату, другой класс нет, и имеет не репрезентативные данные, в результате при обучении сеть учится распознавать первый класс, второй нет, при усреднении результата по всем классам ухудшается качество итогового классификатора. Для предотвращения переобучения используют раннюю остановку обучения, ансамбли, методы оптимизации градиентного спуска, нормализацию, использование прореживающих слоев dropout, а также увеличение объема обучающей выборки [6].

Анализ результатов различных экспериментов показывает, что использование ансамблей моделей СНС может значительно увеличить точность прогнозирования. Это происходит благодаря тому, что каждая модель может обнаруживать определенные характеристики или паттерны в данных, которые могут быть упущены другими моделями [7]. Комбинирование предсказаний всех моделей позволяет снизить вероятность ложных прогнозов и повысить общую точность.

Более того, использование ансамблевых методов может помочь минимизировать эффект переобучения – явления, при котором модель "запоминает" данные обучающего набора и теряет способность к обобщению на новые данные [8]. Ансамблирование позволяет учитывать различные тренды в данных и создавать более устойчивую модель прогнозирования. Принцип работы ансамбля заключается в том, что каждая нейронная сеть обрабатывает входные данные независимо и выдает свой собственный прогноз. Затем, путем агрегации результатов всех нейронных сетей, получается конечный прогноз ансамбля.

На сегодняшний день существует множество методов ансамблирования моделей в зависимости от типа задачи. Исследуемая задача относится к многоклассовой одновариантной классификации изображений. Для распознавания образов применяются нейронные сети со сверточной архитектурой.

Объединение прогнозов из нескольких классификаторов является широко используемой стратегией для повышения точности классификации. Самый простой способ объединения прогнозов – это просто взять среднее арифметическое. Однако этот подход может быть неэффективен, если отдельные классификаторы имеют сильно различающуюся точность [9]. Более эффективным способом объединения прогнозов является использование взвешенного среднего, где каждому классификатору присваивается вес в соответствии с его точностью. Вес классификатора может быть определен на основе его производительности на проверочном наборе данных. Для поиска оптимальных весов можно использовать различные методы оптимизации, такие как случайный поиск, дифференциальный поиск или оптимизация Нелдера-Мида [10]. Оптимальные веса могут зависеть от конкретного набора данных и используемых классификаторов. Другим важным фактором, влияющим на эффективность ансамбля классификаторов, является разнообразие классификаторов. Классификаторы, которые делают схожие ошибки, не будут дополнять друг друга и не улучшат точность ансамбля. Поэтому желательно использовать классификаторы, которые используют разные методы классификации или имеют разные архитектуры [9].

Ввиду того, что основная часть алгоритмов для классификации основывается на поиске экстремума определенной целевой функции, возможно, что классификационное решение окажется вблизи локального экстремума. Чтобы повысить вероятность нахождения глобально-оптимального решения, можно использовать ансамбли моделей, которые объединяют результаты различных базовых классификаторов, построенных на разных выборках исходных данных. Такой подход позволяет искать решение из разных точек, что способствует повышению шансов на получение оптимального результата [11].

Ансамбль базовых алгоритмов вида $a_w(z) = f(b_w(z))$ ($w=1,...,W$) может принимать вид: $a(z) = f(\text{corr}(b_1(z), b_2(z), ..., b_W(z)))$. То есть, это последовательность операций: $\text{bw}: K \rightarrow N$, операции корректировки $\text{corr}: N^W \rightarrow N$ и решающие правила $f: N \rightarrow U$, где K – объекты данных, U – множество размеченных классов, N и U это набор данных, включающий объекты или кортежи данных и соответствующие им метки классов, N это множество оценок [11].

Тогда, например, если у нас имеется 5 независимых алгоритмов, каждый из которых определяет класс изображения с некой вероятностью p_0 , то вероятность после ансамблирования будет равна $p_0^5 + C_5^4 p_0^4 (1-p) + C_5^3 p_0^3 (1-p)^2$.

Существует несколько способов объединения результата в ансамбле [12]:

-простое голосование:

$$a(\vec{x}) = \frac{1}{T} \sum_{t=1}^T a_t(\vec{x}); \quad (1)$$

-взвешенное голосование:

$$a(\vec{x}) = \sum_{t=1}^T \alpha_t a_t(\vec{x}), \quad (2)$$

где α_t можно подбирать линейными методами, $\alpha_t > 0$; $\sum \alpha_t = 1$;

-смешенное голосование:

$$a(\vec{x}) = \sum_{t=1}^T g_t(\vec{x}) a_t(\vec{x}, \vec{\theta}_t), \quad (3)$$

где $g_t(x)$ это функция компетентности, которая способна определить области где более уместен тот или иной алгоритм.

При простом голосовании, выбирается класс, который наиболее соответствует большинству голосов базовых классификаторов. Однако взвешенное голосование отличается от простого голосования в том, что оно присваивает веса голосам базовых классификаторов, учитывая их точность работы. Возможно использование функциональной зависимости как коэффициента веса при сочетании экспертных мнений. Помимо различных методов голосования, смешивание результатов ансамбля может быть выполнено через усреднение, как с учетом веса, так и без него. Например, если результат классификации представлен вероятностями принадлежности объектов классам, а не метками классов. При усреднении без учета веса, итоговое значение ансамбля рассчитывается как простое среднее всех результатов базовых классификаторов. В случае взвешенного усреднения, результаты, полученные от базовых классификаторов, умножаются на соответствующие весовые коэффициенты [11].

Для оптимизации ансамблей нейронных сетей часто применяют метод бустинга (стохастический, адаптивный, градиентный), они заложены в методах адаптивной оптимизации НС, таких как Adagrad, rmsprop, adadelta, метод адаптивной оценки моментов (Adam) [13, 14, 15].

Преимущества использования ансамблей сетей: имеют большую гибкость, возможность достичь более лучших показателей качества моделей за счет вариативности различных алгоритмов, однако использование ансамблей является затратным в плане вычислительных мощностей [11].

Широкое применение ансамблей нейронных сетей возникает в ситуациях, где имеется огромный объем данных, больше 100 тысяч примеров. В данном случае, набор нейронных сетей образует ансамбль, превосходящий случайный выбор в предсказательной точности при данной плотности данных. Эти сети имеют простую структуру, что позволяет быстро обучаться. Их теоретическое основание основано на принципе центральной предельной теоремы в теории вероятностей. Эта теорема утверждает, что последовательность средних значений, вычисленных на основе независимого набора из корня из n случайных величин, имеющих большую дисперсию σ и подчиняющихся нормальному распределению, сходится к нормальному распределению. Ансамбль нейронных сетей использует этот принцип для повышения качества прогнозов [16]:

$$P(O_i|P_j) = \frac{P(P_j|O_i)P(O)}{P(P)}, \quad (4)$$

где, $P(O_i|P_j)$ - это условная вероятность отрицательной погрешности i -ой сети на t -ом шаге обучения при положительной погрешности j -ой сети, $P(P_j|O_i)$ - это условная вероятность положительной погрешности j -ой сети при отрицательной i -ой, $P(P)$, $P(O)$ - сами вероятности появления положительного и отрицательного прогноза.

Путем объединения прогнозов нескольких нейронных сетей и их усреднения можно значительно снизить среднеквадратическое отклонение на \sqrt{n} . Иначе говоря, если суммировать прогнозы отдельных моделей при определенных условиях, то можно уменьшить неопределенность, сопутствующую каждой из них. Таким образом, точность прогнозирования возрастает с увеличением количества использованных нейронных сетей и среднего значения их выходных сигналов. Важно, чтобы ошибки прогнозов отдельных нейронных сетей были статистически независимыми, для достижения этого можно применить различные методы, например, обучение нейронных сетей на разных наборах данных и использование разнообразных моделей [16].

Для достижения высокого качества ансамбля нейронных сетей необходимо обеспечить их полную статистическую независимость. Это означает, что каждая сеть должна обучаться на различных и независимых наборах данных. Однако, усреднение результатов от этих сетей может не привести к существенному снижению дисперсии выходного сигнала. Поэтому в таких случаях часто используется метод взвешенного голосования. Путем присвоения различных весов каждой сети, можно достичь оптимального результата. Этот подход позволяет учесть вклад каждой сети и создать более точный и надежный ансамбль [16-20]:

$$\hat{y}_R(\hat{y}) = \frac{1}{n} \sum_{i=1}^n a_i \hat{y}_i(\mathbf{x}) + b, \quad (5)$$

где $y_i(\mathbf{x})$ это выход нейронной сети, $y_R(y)$ это вероятность принадлежности к классу, y это вектор вероятностей принадлежности к классами, b это весовой коэффициент.

В этом случае дисперсия будет определяться по формуле:

$$\sigma_R^2 = \frac{1}{n} \left(\sum_{i=1}^n a_i^2 \sigma_i^2 + 2 \sum_{i < j} a_i a_j r_{ij} \sigma_i \sigma_j \right), \quad (6)$$

где rij это коэффициент корреляции между выходами от нескольких нейронных сетей.

Существует необходимость в использовании многоуровневой аппроксимации для уменьшения дисперсии результирующего прогноза на основе коррелирующих выходных сигналов i -ой и j -ой нейронных сетей на реальных наборах данных. Для этого важно создать многоуровневую архитектуру ансамбля с 2-3 уровнями, где каждый уровень будет аппроксимировать прогнозы предыдущего уровня. Дополнительным условием является минимальная корреляция выходных сигналов нижнего уровня. Веса при этом задаются самостоятельно и с помощью специальных методов осуществляется поиск оптимальных весов, показывающих вклад каждой модели в итоговый прогноз [\[16\]](#).

Для проведения экспериментов по классификации изображений был выбран открытый источник данных по дерматологическим заболеваниям ISIC HAM10000 [\[21\]](#). Он содержит 10015 изображений заболеваний кожи. Помимо загрузки изображений, загружен файл csv с размеченными данным по изображениям, это возраст пациента, локализация образования, пол, код изображения и целевой класс это диагноз. Данный набор был выбран так как имеется полностью размеченная выборка, данные являются репрезентативными и является самым большим набором в этой области. Исследование велось на языке python с аппаратным графическим ускорителем, с использованием фреймворков машинного обучения и разработки нейронных сетей, таких как keras, tensorflow, sklearn.

Результаты и обсуждение

Для создания ансамбля сетей было принято решение обучить несколько сверточных нейронных сетей с различными параметрами и архитектурой на разных обучающих выборках. Мы считали наши изображения и преобразовали их в вектора тензоров с помощью библиотеки `numpy`, при этом изменив размерность входного изображения, чтобы снизить нагрузку на вычислительные мощности. Классы были очень не сбалансированы, поэтому мы провели очистку данных самого большого по объему класса и аугментировали данные по количеству максимального класса, путем вращений, изменения цвета, обрезки изображений, чтобы добиться сбалансированности целевых классов. Затем мы считали полученный датафрейм с разметками классов и объединили с полученными тензорами изображений по коду изображения.

Предварительная обработка данных также очень важна для повышения качества будущего классификатора, поэтому в ходе анализа, полученный набор данных был исследован на аномалии, преобразование категориальных переменных, трансформацию данных, удаление дубликатов [\[22, 23\]](#). С помощью метода `train_test_split` мы разделили выборку на обучающую и тестовую, где 80% обучающей выборки, 20% на тестовую. По полученным данным провели нормализацию и стандартизацию данных. Затем полученную обучающую выборку мы также разделили на обучающую и валидационную в размере 90% обучающей и 10% валидационной. Полученную обучающую выборку мы также разделили на три независимых набора с вероятностью выбора 20% для каждой модели.

Для минимизации корреляции между прогнозами отдельных моделей мы их обучили на разных независимых выборках. Первая модель спроектирована согласно методу трансферного обучения [\[24\]](#), мы подгрузили архитектуры и веса нескольких предварительно обученных моделей на ImageNet, с разной архитектурой с целью минимизации дисперсии погрешностей, показавшие хорошие результаты в

распознавании образов, такие как InceptionV3, InceptionResNetV2, VGG16 от фреймворка keras и tensorflow [25, 26, 27]. Задали размерность входного изображения 75x75 пикселей ввиду ограниченности вычислительных мощностей, после чего мы заморозили слои предобученных моделей и изменили последний слой под количество целевых классов, в нашей задаче их 7 [28, 29]. На рисунке 2 показана схема реализации трансферного обучения.

```
from tensorflow.keras.applications import InceptionV3, InceptionResNetV2, VGG16
model1 = InceptionResNetV2(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
model2 = InceptionV3(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
model3 = VGG16(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
model1.trainable = False
model2.trainable = False
model3.trainable = False
model1_v1 = model1.output
model1_v2=GlobalAveragePooling2D()(model1_v1)
model1_v3 = Dense(1024, activation='relu')(model1_v2)
pred1 = Dense(7, activation='softmax')(model1_v3)
model_final1 = Model(inputs=model1.input, outputs=pred1)
model2_v1 = model2.output
model2_v2=GlobalAveragePooling2D()(model2_v1)
model2_v3 = Dense(1024, activation='relu')(model2_v2)
pred2 = Dense(7, activation='softmax')(model2_v3)
model_final2 = Model(inputs=model2.input, outputs=pred2)
model3_v1 = model3.output
model3_v2=GlobalAveragePooling2D()(model3_v1)
model3_v3 = Dense(1024, activation='relu')(model3_v2)
pred3 = Dense(7, activation='softmax')(model3_v3)
model_final3 = Model(inputs=model3.input, outputs=pred3)
model_final1.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
model_final2.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
model_final3.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

Рисунок 2. Реализация трансферного обучения

Полученные модели мы скомпилировали, в качестве оптимизации мы выбрали оптимизатор Adam, в качестве функции потерь categorical_crossentropy, в качестве метрики выбрали accuracy. Каждую модель мы обучили на 40 эпохах и размером батча в 10 пакетов на разных выборках. Каждая модель показала разный результат, худший результат показала модель InceptionV3 в 53% точности, поскольку она имеет самую глубокую архитектуру и большее количество параметров, поэтому обучение на малом объеме данных показывает низкое качество. Для реализации объединения результатов простым голосованием мы применили слой Average для усреднения выходного сигнала от каждой модели. На рисунках 3, 4 показана реализация метода объединения результата.

```
models = [model_final1, model_final2, model_final3]
input = Input(shape=(75, 75, 3), name='input')
outputs = [model(input) for model in models]
x = Average()(outputs)
x = Dropout(0.5)(x)
output = Dense(7, activation='softmax', name='output')(x)
conc_model = Model(input, output, name='Model')
conc_model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

Рисунок 3. Реализация простого усреднения предобученных моделей

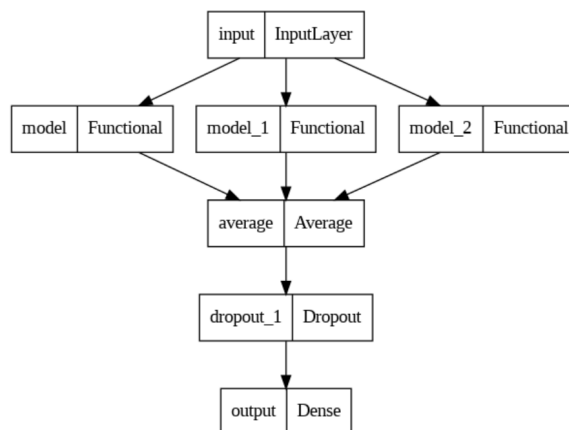


Рисунок 4. Архитектура ансамбля с простым усреднением

Мы создали новую архитектуру модели, добавили новый слой Input для входного изображения, от каждой модели получили выходные модернизированные слои и передали их в усредняющий слой Average. С целью предотвращения переобучения добавили прореживающий слой dropout и выходной полносвязный слой с 7 целевыми классами и функцией активации softmax. Полученная архитектура скомпилирована и обучена аналогично. На 50 эпохе обучения точность на обучающей выборке составляла 71%, точность на валидационной выборке 76%. На рисунке 5 показан процесс обучения. Точность на тестовой выборке составила 74% рисунок 6.

```
Epoch 45/50
1512/1512 [=====] - 74s 49ms/step - loss: 0.8227 - accuracy: 0.7094 - val_loss: 0.7243 - val_accuracy: 0.7625
Epoch 46/50
1512/1512 [=====] - 77s 51ms/step - loss: 0.8260 - accuracy: 0.7093 - val_loss: 0.7242 - val_accuracy: 0.7625
Epoch 47/50
1512/1512 [=====] - 72s 48ms/step - loss: 0.8353 - accuracy: 0.7060 - val_loss: 0.7242 - val_accuracy: 0.7625
Epoch 48/50
1512/1512 [=====] - 72s 48ms/step - loss: 0.8266 - accuracy: 0.7095 - val_loss: 0.7241 - val_accuracy: 0.7625
Epoch 49/50
1512/1512 [=====] - 75s 50ms/step - loss: 0.8185 - accuracy: 0.7095 - val_loss: 0.7241 - val_accuracy: 0.7625
Epoch 50/50
1512/1512 [=====] - 76s 50ms/step - loss: 0.8193 - accuracy: 0.7138 - val_loss: 0.7241 - val_accuracy: 0.7625
```

Рисунок 5. Процесс обучения ансамбля с простым усреднением

```
accuracy = accuracy_score(np.argmax(y_test2, axis=1), y_pred)
precision = precision_score(np.argmax(y_test2, axis=1), y_pred, average='macro')
recall = recall_score(np.argmax(y_test2, axis=1), y_pred, average='macro')
f1 = f1_score(np.argmax(y_test2, axis=1), y_pred, average='macro')
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
```

```
Accuracy: 0.7412
Precision: 0.7411
Recall: 0.7440
F1-score: 0.7421
```

Рисунок 6. Оценка качества модели ансамбля на тестовой выборке

Однако мы видим, что на последних эпохах изменений в качестве обучения не происходило, что говорит о переобучении модели, поскольку обучающая выборка составляла меньше 10 тысяч примеров, что для обучения нейронных сетей с очень глубокой архитектурой является недостаточным, поэтому разработка собственной архитектуры с минимальным количеством уровней свертки является более оптимальным для этой задачи, причем количество классификаторов в ансамбле должно быть больше 3 и он должен иметь иерархическую архитектуру. Использование слабой архитектуры мы применим совместно с алгоритмом формирования средневзвешенного ансамбля [14]. В ходе экспериментальных исследований был выработан алгоритм к формированию ансамбля СНС.

Первым шагом мы спроектировали и оценили качество эффективной архитектуры СНС как базовой модели. На рисунке 7 представлена архитектура эффективной СНС.

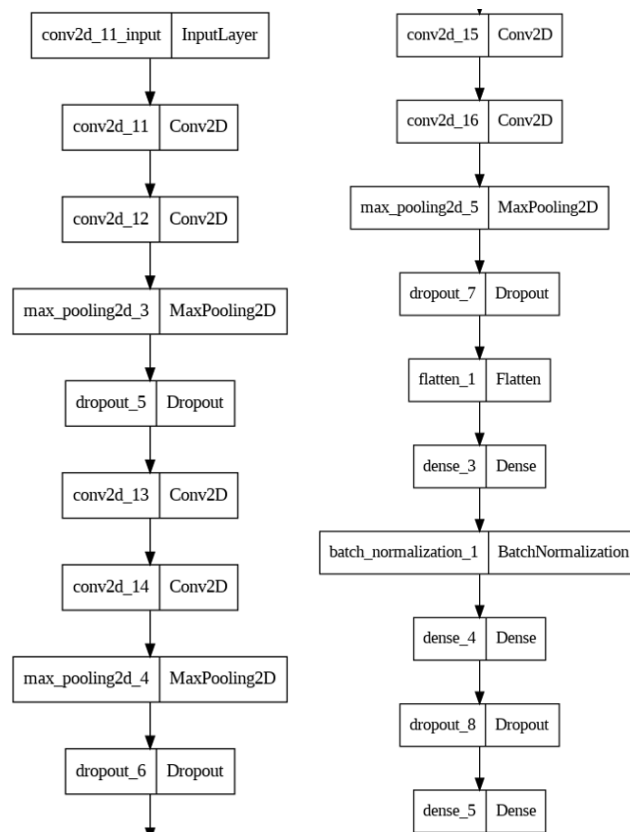


Рисунок 7. Эффективная архитектура СНС

Данная модель состоит из трех сверточных уровней, в каждом из которых имеется два сверточных слоя, слой пулинга и слой прореживания, экспериментальным путем подобраны гиперпараметры сети, количество фильтров в слое с увеличением на каждом следующем слое в 2 раза, размер ядра начиная с 7x7 и последующем применении ядра 3x3, функции активации, чередование слоев, в конце добавлен один полносвязный слой, слой нормализации flatten, прореживания dropout и один выходной полносвязный слой dense для классификации 7 целевых классов. В ходе исследования выработаны рекомендации к формированию архитектуры эффективного нейросетевого классификатора [\[30-33\]](#):

-необходима предварительная обработка данных, анализ сбалансированности классов, в случае малого объема выборки необходимо обогащение;

-чем выше размерность входного изображения, тем большее количество уровней свертки необходимо использовать;

-также важен выбор оптимизатора при компиляции сети, если эпох обучения задано много, необходимо использовать оптимизатор SGD, иначе Adam;

-должно быть чередование слоев свертки и слоя пулинга, количество слоев свертки задается оптимально от 1 до 2 в зависимости от сложности входного изображения, для огромных объемов данных, больше 100 тысяч примеров, можно использовать количество сверток больше 3, количество слоев пулинга от 0 до 1 на одном уровне свертки с размером ядра (2,2) и функцией MaxPooling2D;

- на последних слоях необходимо уменьшать количество полносвязных слоев, оптимальным является один полносвязный слой и один выходной полносвязный слой;
- для уменьшения вероятности переобучения необходимо использовать слои dropout и BatchNormalization;
- важным параметром слоя свертки является размер ядра фильтра свертки, которое может принимать значения (3,3), (5,5), (7,7), чем сложнее входное изображение, тем выше необходимо задавать размер фильтра сверточного слоя;
- размер входного изображения input_shape на входном слое должен быть квадратным;
- параметры padding и stride сверточного слоя должны быть целочисленными и равны значениям предыдущего слоя, если он не был тоже сверточным;
- количество фильтров filters в сверточном слое задается оптимальным образом, чем выше размер ядра фильтра, тем большее число задается, и с каждым последующим уровнем свертки увеличивается в 2 раза.

В ходе исследования выявлены важные параметры архитектуры нейронной сети, которые влияют на ее эффективность в процессе обучения: размер фильтра сверточного слоя, тип чередования слоев сети, а также количество слоев свертки на одном уровне, количество фильтров, параметр filters, используемых в сверточном слое, количество уровней свертки.

Следующим шагом мы обучили модель, она показала достаточно хороший результат, практически 90% точности на обучающей и 78% на тестовой выборке рисунок 8. Данную архитектуру мы применили для ансамбля с оптимизированным поиском весов и средневзвешенным усреднением.



Рисунок 8. График изменения точности в процессе обучения эффективной СНС

Библиотека SciPy имеет большое количество методов оптимизации, например, метод differential_evolution [\[34\]](#), он находит глобальный минимум многомерной функции, при этом не использует градиентные методы и носит случайный характер, на вход функции подается функция для оценки набора весов, границы весов, количество моделей в ансамбле, тестовый набор X и y, количество итераций поиска, применяя данный метод можно минимизировать дисперсию выходного сигнала моделей ансамбля.

Реализована функция обучения моделей, в которой задана архитектура и для каждой модели формируется отдельный набор данных для обучения, чтобы обеспечить

независимость моделей ансамбля во время обучения. Обучение таких моделей происходит на разных выборках, сами архитектуры должны иметь различающиеся параметры, что повысит адаптивность к изменению входных данных ансамбля. Следующим шагом реализованы функции оценки качества ансамбля и набора весов [35]. На рисунке 9 показаны основные функции алгоритма, реализованные на python.

```
def ensemble_predict(m, w, x_test4):
    y = [model.predict(x_test4) for model in m]
    y = array(y)
    s = tensordot(y, w, axes=(0,0))
    res = argmax(s, axis=1)
    return res
def evaluate_ensemble(m, w, x_test4, y_test2):
    y = ensemble_predict(m, w, x_test4)
    return accuracy_score(y_test2, y)
def normalize(w):
    res = norm(w, 1)
    if res == 0.0:
        return w
    return w / res
def loss_function(w, m, x_test4, y_test2):
    norm = normalize(w)
    return 1.0 - evaluate_ensemble(m, norm, x_test4, y_test2)
```

Рисунок 9. Реализация основных функций метода взвешенной оценки

Для реализации алгоритма поиска оптимальных весов необходима функция оптимизации, для этого была создана функция `loss_function`, функция оценки весов получает на вход вектор весов `w`, количество моделей ансамбля `m`, а также тестовые `X_test` и `y_test`, в результате функции полученный вес нормализуется с помощью второй функции `normalize`, в которой с помощью `numpy.norm` нормализуется значение веса, исходный вес делится на нормализованную величину, по итогу функции `loss` рассчитывается доля истинно отрицательных ответов и рассчитывается как 1 минус точность положительного прогноза, для этого вызывается функция `evaluate_ensemble`, в которую передается количество моделей, нормализованная величина веса и тестовый пример, в этой функции рассчитывается точность положительного прогноза с помощью `accuracy_score`, в которую передается фактический класс из тестового набора и прогнозируемый класс, для определения прогнозируемого класса вызывается функция `ensemble_predictions`, в которой для каждой модели делается прогноз на тестовом наборе с помощью метода `predict`, формируется массив вероятностей принадлежности к классам, затем с помощью метода `tensordot` формируется скалярное тензорное произведение по осям вероятностей на полученный вес, то есть на вход метод получает матрицу размером 5x7, 5 моделей ансамбля и 7 целевых классов, по каждому классу модель сформировала вероятность принадлежности, метод транспонирует матрицу, тогда по горизонтали получается 7 по вертикали 5, затем каждый вектор матрицы умножает на заданный вес, после чего суммирует взвешенные сигналы моделей по столбцам, тем самым получается 7 взвешенных вероятностей принадлежности от каждой модели ансамбля, из массива вероятностей принадлежности с учетом весов выбирается максимальное значение и берется его индекс с помощью метода `argmax`. В результате мы получаем прогнозируемый класс с учетом заданных весов.

Функция оценки весов в нашем случае является функцией потерь и определяет долю истинно отрицательных результатов, которые ухудшают качество и определяется как 1 минус точность положительного прогноза. На рисунке 10 показана блок-схема алгоритма

поиска оптимальных весов.

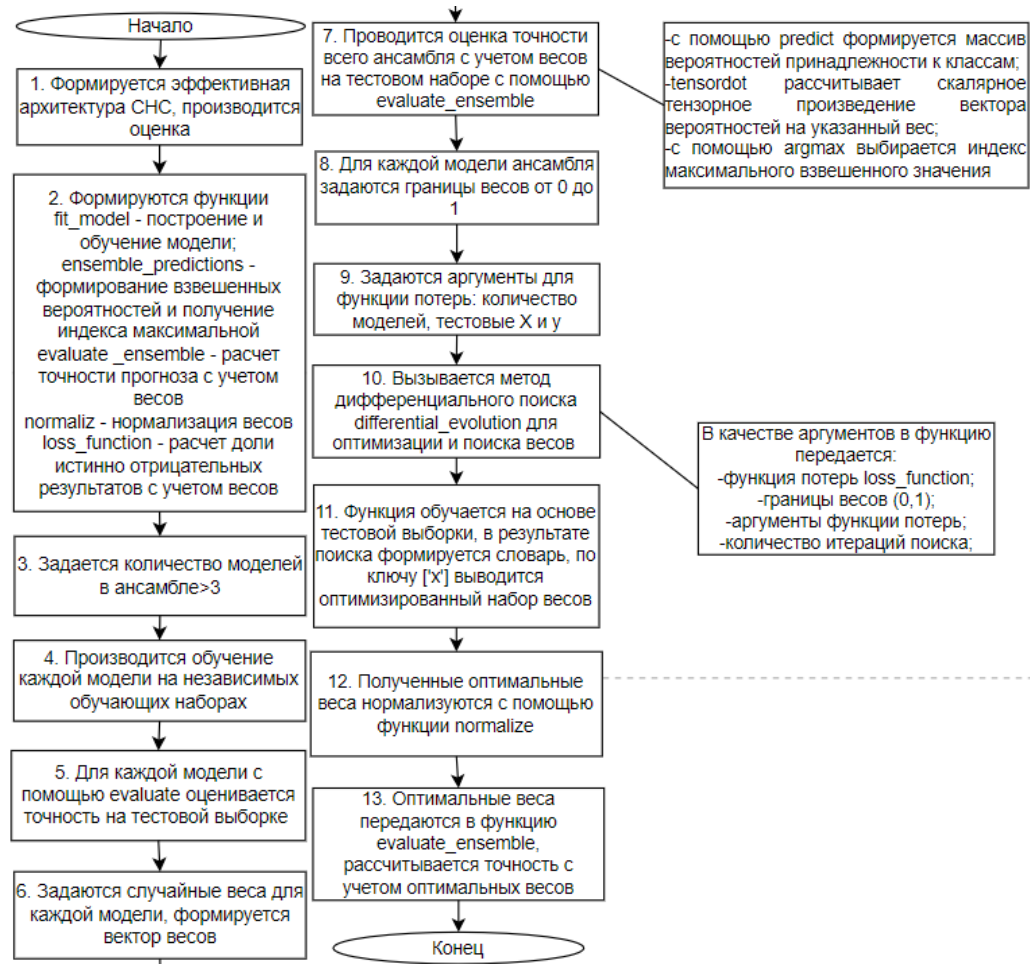


Рисунок 10. Блок-схема алгоритма формирования взвешенного ансамбля

Задается количество моделей в ансамбле, каждая модель обучается на своем независимом наборе данных, с помощью метода evalute каждая модель оценивается и рассчитывается ее точность. Формируется список весов для каждой модели, с помощью функции ensemble_predictions оценивается точность ансамбля с учетом заданных весов, веса на начальном этапе задаются случайными, задаются границы весов от 0 до 1, указываются аргументы функции оптимизации весов, это количество моделей, тестовый набор X и y, после чего вызывается функция оптимального поиска весов differential_evolution, в которую передается функция оптимизации, ограничение весов, аргументы и задается количество итераций поиска. По ключу X мы получаем оптимальный набор весов для каждой модели вида (0.01, 0.2, 0.11, 0.49, 0.12), далее оцениваем качество ансамбля по уже оптимальным весам. Мы апробировали результаты алгоритма на тестовом наборе данных. В результате алгоритм формирует 5 сверточных сетей и оценивает их точность таблица 1.

Модель	Точность
CNN 1	0,80
CNN 2	0,82
CNN 3	0,81
CNN 4	0,83
CNN 5	0,81

Таблица 1. Результаты взвешенных оценок моделей ансамбля

В таблице 2 представлено сравнение полученных оценок точности каждой модели.

Метод	Accuracy	Precision	Recall	F1 мера
Ансамбль из трех предварительной обученных моделей VGG16, InceptionV3, InceptionResNetV2 на основе простого усреднения результатов моделей	0,74	0,74	0,74	0,74
Одиночная СНС с эффективной архитектурой согласно алгоритму	0,78	0,78	0,78	0,77
Ансамбль из 5 эффективных СНС на основе взвешенной оценки результатов согласно алгоритму	0,83	-	-	-

Таблица 2. Оценка качества полученных моделей

Модель формирует результат точности с учетом взвешенных оценок, оценка точности на оптимальных весах равна 0,83, что выше, чем у модели на основе ансамбля глубоких предварительно обученных сетей. Алгоритм находит оптимальные веса и показывает весовую значимость каждой модели ансамбля, полученные веса передаются в функцию `ensemble_predictions`, тем самым прогнозируется точность с учетом оптимальных весов, на основе полученных весов формируется оценка точности объединяющего классификатора, которая достигает 83%, что на 5% выше чем, у одиночной эффективной архитектуры СНС и на 9% выше, чем у ансамбля из глубоких предобученных моделей с простым усреднением результатов.

Работу модели мы проверили на тестовом примере, мы подгрузили тестовое изображение дерматологического заболевания меланоцитарного невуса и дерматофибромы, уменьшили размерность к (75, 75), поскольку модель обучалась на этой размерности, затем преобразовали в тензорное представление с помощью библиотеки `patru.asarray`, чтобы добиться 0 среднего значения и 1 дисперсии полученный вектор тензоров нормализовали путем вычета среднего и деления на стандартное отклонение. На рисунке 11 показана программная реализация Загрузки и обработки изображения.

```
f_name = '/content/drive/MyDrive/Colab Notebooks/ISIC_0024319.jpg'
image = np.asarray(Image.open(f_name).resize((75,75)))
plt.imshow(image)
plt.show()
img = np.asarray(image.tolist())
x__mean = np.mean(img)
x__std = np.std(img)
img = (img - x__mean)/x__std
img = np.asarray(img)[None, ...]
img.shape
```

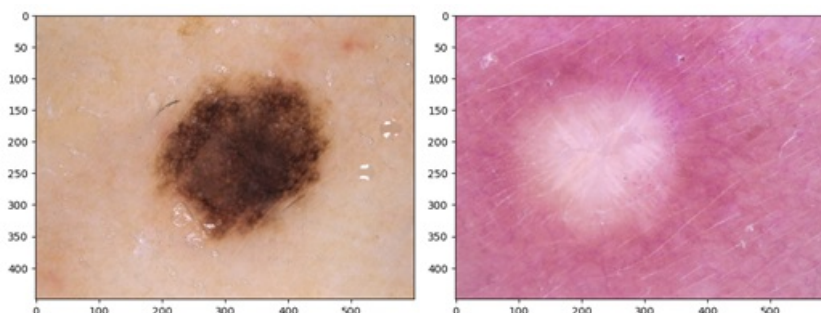


Рисунок 11. Загрузка и обработка входного изображения

Затем мы написали функционал на python для прогнозирования класса заболевания и отображения гистограммы распределения вероятностей по целевым классам. На рисунке

12 показан программная реализация для двух тестовых фотоснимков.

```
import matplotlib.pyplot as plt
lesion_type_idx_dict = {
    4: 'Melanocytic nevi',
    5: 'Melanoma',
    2: 'Benign keratosis-like lesions ',
    1: 'Basal cell carcinoma',
    0: 'Actinic keratoses',
    6: 'Vascular lesions',
    3: 'Dermatofibroma'
}
print(conc_model.predict(img))
classes=np.argmax(conc_model.predict(img), axis=-1)
print(lesion_type_idx_dict.get(classes[0]))
pred=conc_model.predict(img)
bur=pred[0]
t= {i: bur[i] for i in range(len(bur))}
t=pd.DataFrame(list(t.items()), columns=['Key', 'Values'])
t['Description'] = t['Key'].map(lesion_type_idx_dict)
plt.barh(t['Description'],t['Values'], color='blue')

1/1 [=====] - 0s 236ms/step
[[0.01158232 0.01232567 0.03795601 0.00943617 0.84017426 0.04994805
 0.03857748]]
1/1 [=====] - 0s 292ms/step
Melanocytic nevi
```

```
import matplotlib.pyplot as plt
lesion_type_idx_dict = {
    4: 'Melanocytic nevi',
    5: 'Melanoma',
    2: 'Benign keratosis-like lesions ',
    1: 'Basal cell carcinoma',
    0: 'Actinic keratoses',
    6: 'Vascular lesions',
    3: 'Dermatofibroma'
}
print(conc_model.predict(img))
classes=np.argmax(conc_model.predict(img), axis=-1)
print(lesion_type_idx_dict.get(classes[0]))
pred=conc_model.predict(img)
bur=pred[0]
t= {i: bur[i] for i in range(len(bur))}
t=pd.DataFrame(list(t.items()), columns=['Key', 'Values'])
t['Description'] = t['Key'].map(lesion_type_idx_dict)
plt.barh(t['Description'],t['Values'], color='blue')

1/1 [=====] - 0s 435ms/step
[[0.01785488 0.01877308 0.00871841 0.94005877 0.00505726 0.00584777
 0.0036898 ]]
1/1 [=====] - 0s 373ms/step
Dermatofibroma
```

Рисунок 12. Программная реализация функции прогноза и распределения вероятностей

В ходе разработки был создан словарь с закодированными целевыми классами на этапе предварительной обработки и трансформации датасета метаданных. Закодированные классы представлены в словаре, с помощью метода модели predict мы предсказали вероятности принадлежности фотоснимка к тому или иному классу, для первого мы видим, что наибольшая вероятность равна 0,84 с индексом 4, для второго 0,94 с индексом 3, с помощью метода argmax мы получили индекс максимального значения вероятности в массиве, соответственно равны 4 и 3, по полученному индексу и равному ключу словаря определили описание диагноза, для первого модель точно предсказала меланоцитарный невус, для второй дерматофибром с вероятностями 84% и 94% соответственно. Для конечного пользователя важно интерпретировать результаты нейросети, как она пришла к этому исходу, поскольку зачастую она является моделью "черного ящика", поэтому гистограмма по распределению вероятностей покажет почему модель пришла к этому результату, однако более информативным для интерпретации результатов нейросетей является построение тепловых карт GradCam, показывающих на каких ключевых областях изображения были сделаны выводы. Для реализации гистограммы мы сделали прогноз и получили список индексов и значений вероятностей, создали словарь, где ключ это индекс, а значение это вероятность для каждого класса, для удобства создали датафрем с ключом и значением вероятности. С помощью метода датафрема map мы по индексу, равному ключу описательного словаря, получили наименование классов. С помощью библиотеки matplotlib.pyplt мы построили гистограммы распределения вероятностей рис. 13.

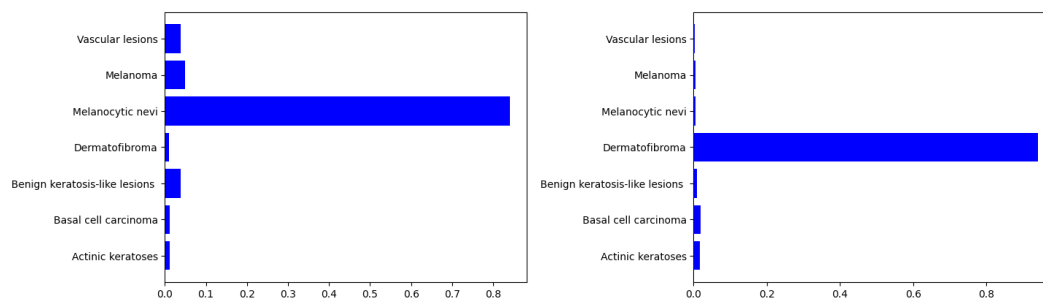


Рисунок 13. Гистограммы распределения вероятностей

На гистограммах видно, какой класс является наиболее вероятным. Полученные результаты исследования и модели будут использованы для разработки

интеллектуальной адаптивной рекомендательной системы, способной перенастраиваться и переобучаться для детектирования новых случаев. На рисунке 14 показан интерфейс окна диагностики по фотоснимкам.

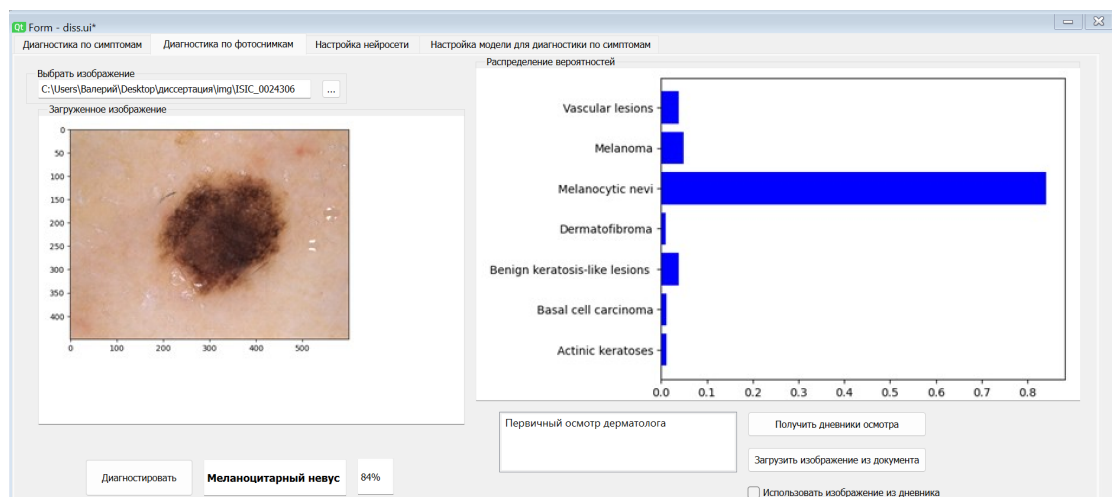


Рисунок 14. Интерфейс окна диагностики по фотоснимкам

Таким образом, мы выработали алгоритм к формированию эффективной архитектуры СНС на основе ансамбля с средневзвешенным объединением результатов, выработаны рекомендации к формированию эффективной базовой архитектуры СНС, проведено исследование ансамбля с применением предварительно обученных нейросетей на ImageNet с глубокой архитектурой на основе простого усреднения и с применением метода трансферного обучения на малом объеме обучающей выборки, в результате модель переобучилась и показала низкий результат по сравнению с другими моделями, эту проблему удалось решить обогащением выборки и равномерным распределением обучающей и тестовой выборки по каждому классу.

Заключение

В ходе исследования мы реализовали несколько ансамблей СНС, с применением алгоритмов усреднения и объединения результатов моделей, первый алгоритм на основе простого арифметического усреднения моделей ансамбля оказался не оптимальным в применении к решаемой задаче, поскольку использует НС с очень глубокой архитектурой, обучение производилось на небольшой выборке, поэтому произошло переобучение сети, вследствие чего понизилось качество итогового классификатора, точность достигла 74% на тестовой выборке. Вторая модель реализована на основе ансамбля моделей с эффективной архитектурой НС и с средневзвешенным усреднением результата, она показала результат лучше, чем предыдущая модель и достигла точности в 83%, что на 9% выше. Таким образом, мы экспериментальным путем апробировали результаты построенных моделей, выработали алгоритм к формированию эффективного ансамбля СНС с применением методов усреднения, исследовали влияние переобучения сети на конечный результат, сформировали рекомендации к построению эффективного базового классификатора СНС и построили на основе алгоритмов архитектуры и модели СНС, полученные результаты будут использованы для разработки интеллектуальной системы поддержки принятия врачебных решений.

Библиография

1. Thoma M. Analysis and optimization of convolutional neural network architectures, 2017.

2. Cruz Y. J. et al. Ensemble of convolutional neural networks based on an evolutionary algorithm applied to an industrial welding process //Computers in Industry. – 2021. – Т. 133. – С. 103-530.
3. Yang S. et al. An ensemble classification algorithm for convolutional neural network based on AdaBoost //2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS). – IEEE, 2017. – С. 401-406.
4. Basili V. R., Briand L. C., Melo W. L. A validation of object-oriented design metrics as quality indicators //IEEE Transactions on software engineering. – 1996. – Т. 22. – №. 10. – С. 751-761.
5. Нейронные сети. Переобучение-что это и как этого избежать, критерии останова обучения. [Электронный ресурс]. URL: https://proproprogs.ru/neural_network/pereobuchenie-chto-eto-i-kak-etogo-izbezhat-kriterii-ostanova-obucheniya (дата обращения 09.02.2024).
6. Воронецкий Ю. О., Жданов Н. А. Методы борьбы с переобучением искусственных нейронных сетей // Научный аспект. 2019. №2. [Электронный ресурс] URL: <https://na-journal.ru/2-2019-tehnicheskie-nauki/1703-metody-borby-s-pereobucheniem-iskusstvennyh-neironnyh-setei> (дата обращения: 10.02.2024).
7. Li C. et al. Improving forecasting accuracy of daily enterprise electricity consumption using a random forest based on ensemble empirical mode decomposition // Energy. – 2018. – Т. 165. – С. 1220-1227.
8. Omisore O. M. et al. Weighting-based deep ensemble learning for recognition of interventionalists' hand motions during robot-assisted intravascular catheterization // IEEE Transactions on Human-Machine Systems. – 2022. – Т. 53. – №. 1. – С. 215-227.
9. Ансамблирование моделей нейронных сетей с использованием библиотеки Keras. [Электронный ресурс]. URL:https://se.moevm.info/lib/exe/fetch.php/courses:artificial_neural_networks:pr_8.pdf (дата обращения 11.02.2024).
10. Метод оптимизации Нелдера — Мида. Пример реализации на Python. [Электронный ресурс]. URL:<https://habr.com/ru/articles/332092/> (дата обращения 09.02.2024).
11. Ключева И. А. Методы и алгоритмы ансамблирования и поиска значений параметров классификаторов. [Электронный ресурс]. URL:https://dissov.pnzgu.ru/files/dissov.pnzgu.ru/2021/tech/klyueva/dissertaciya_klyuevoy_i_a_.pdf (дата обращения 08.02.2024).
12. Микрюков, А. А. Классификация событий в системах обеспечения информационной безопасности на основе нейросетевых технологий / А. А. Микрюков, А. В. Бабаш, В. А. Сизов // Открытое образование. – 2019. – Т. 23. № 1. – С. 57-63.
13. Gizluk D. Adaptive optimization methods. // Neural networks are simple (part 7). 2020. №7. [Электронный ресурс]. URL:<https://www.mql5.com/ru/articles/8598#para21> (дата обращения: 10.02.2024).
14. Mason L. et al. Boosting algorithms as gradient descent //Advances in neural information processing systems. – 1999. – Т. 12.
15. Zaheer R., Shaziya H. A study of the optimization algorithms in deep learning //2019 third international conference on inventive systems and control (ICISC). – IEEE, 2019. – С. 536-539.
16. Староверов Б. А., Хамитов Р. Н. Реализация глубокого обучения для прогнозирования при помощи ансамбля нейронных сетей //Известия Тульского государственного университета. Технические науки. – 2023. – №. 4. – С. 185-189.
17. Onan A., Korukoğlu S., Bulut H. A multiobjective weighted voting ensemble classifier

- based on differential evolution algorithm for text sentiment classification // Expert Systems with Applications. – 2016. – Т. 62. – С. 1-16.
18. Kim H. et al. A weight-adjusted voting algorithm for ensembles of classifiers // Journal of the Korean Statistical Society. – 2011. – Т. 40. – №. 4. – С. 437-449.
19. Yao X., Islam M. M. Evolving artificial neural network ensembles // IEEE Computational Intelligence Magazine. – 2008. – Т. 3. – №. 1. – С. 31-42.
20. Anand V. et al. Weighted Average Ensemble Deep Learning Model for Stratification of Brain Tumor in MRI Images //Diagnostics. – 2023. – Т. 13. – №. 7. – С. 1320.
21. The International Skin Imaging Collaboration. [Электронный ресурс].-URL: <https://www.isic-archive.com> (дата обращения 12.02.2024).
22. Alexandropoulos S. A. N., Kotsiantis S. B., Vrahatis M. N. Data preprocessing in predictive data mining // The Knowledge Engineering Review. – 2019. – Т. 34. – С. e1.
23. García S., Luengo J., Herrera F. Data preprocessing in data mining. – Cham, Switzerland: Springer International Publishing, 2015. – Т. 72. – С. 59-139.
24. Liang G., Zheng L. A transfer learning method with deep residual network for pediatric pneumonia diagnosis // Computer methods and programs in biomedicine. – 2020. – Т. 187. – С. 104-964.
25. InceptionV3. [Электронный ресурс].-URL: <https://keras.io/api/applications/inceptionv3/> (дата обращения 13.02.2024).
26. InceptionResNetV2. [Электронный ресурс]. URL: <https://keras.io/api/applications/inceptionresnetv2/> (дата обращения 13.02.2024).
27. VGG16. [Электронный ресурс]. URL: <https://keras.io/api/applications/vgg/#vgg16-function> (дата обращения 13.02.2024).
28. Щетинин Е. Ю. О некоторых методах сегментации изображений с применением свёрточных нейронных сетей // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем. – 2021. – С. 507-510.
29. Rosebrock A. Change input shape dimensions for fine-tuning with Keras. // AI & Computer Vision Programming. 2019. [Электронный ресурс]. URL:<https://pyimagesearch.com/2019/06/24/change-input-shape-dimensions-for-fine-tuning-with-keras/> (дата обращения 14.02.2024).
30. Костин К. А. и др. Адаптивный классификатор патологий для компьютерной диагностики заболеваний с использованием сверточных нейронных сетей по медицинским изображениям и видеоданным: магистерская диссертация по направлению подготовки: 01.04. 02-Прикладная математика и информатика. – 2017.
31. A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Proceedings of Advances in Neural Information Processing Systems 25 (NIPS 2012), 2012, Pp. 1097-1105.
32. Wang J., Lin J., Wang Z. Efficient hardware architectures for deep convolutional neural network // IEEE Transactions on Circuits and Systems I: Regular Papers. – 2017. – Т. 65. – №. 6. – pp. 1941-1953.
33. Phung V. H., Rhee E. J. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets //Applied Sciences. – 2019. – Т. 9. – №. 21. – С. 4500.
34. The differential evolution method. [Электронный ресурс]. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.differential_evolution.html (дата обращения: 13.02.2024).

35. Как разработать средневзвешенный ансамбль для глубоких обучающих нейронных сетей. // 2018. [Электронный ресурс]. URL: <https://machinelearningmastery.ru/weighted-average-ensemble-for-deep-learning-neural-networks/#> (дата обращения: 13.02.2024)

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Рецензируемая работа посвящена исследованию и разработке алгоритмов формирования эффективного ансамбля сверточных нейронных сетей для классификации изображений путем решения задачи многоклассовой одновариантной классификации изображений на основе обучения модели с учителем на размеченном наборе данных. Методология исследования базируется на применении кибернетического подхода и концепции искусственного интеллекта к распознаванию образов с использованием искусственных нейронных сетей, созданием ансамблей моделей машинного обучения. Эксперименты по классификации изображений были проведены на материалах открытого источника данных по дерматологическим заболеваниям, представленных в формате csv, исследование проведено на языке python с аппаратным графическим ускорителем и использованием фреймворков машинного обучения и разработки нейронных сетей (keras, tensorflow, sklearn).

Актуальность работы авторы справедливо связывают с тем, что для повышения точности прогноза и уменьшения ошибок можно использовать ансамбли моделей нейронных сетей, которые находят широкое применение в разных сферах, включая распознавание изображений в интеллектуальных системах поддержки принятия врачебных решений.

Научная новизна рецензируемого исследования, по мнению рецензента состоит в разработке алгоритмов ансамблевых моделей сверточных нейронных сетей с применением методов усреднения и рекомендациях по применению полученных результатов для разработки интеллектуальной системы поддержки принятия врачебных решений.

В тексте статьи выделены следующие разделы: Введение, Материалы и методы, Результаты и обсуждение, Заключение, Библиография.

Во введении обоснована актуальность темы, отмечены преимущества и недостатки ансамблирования в машинном обучении. Далее обозначена проблема переобучения в использовании глубоких нейронных сетей на малом объеме данных – явление, когда сеть очень хорошо прогнозирует на обучающей выборке и плохо на тестовой, в результате чего наблюдается большая разница показателя качества на обучающей и валидационной выборке. В публикации рассмотрен принцип работы ансамбля, заключающийся в том, что каждая нейронная сеть обрабатывает входные данные независимо и выдает свой собственный прогноз, а затем, путем агрегации результатов всех нейронных сетей, получается конечный прогноз, в котором индивидуальные ошибки компенсируются, а общая производительность системы улучшается. Приведены несколько способов объединения результата в ансамбле: простое голосование, взвешенное голосование, смешанное голосование; представлено сравнение полученных оценок точности трех моделей. Статья иллюстрирована двумя таблицами, 14 рисунками, содержит 6 формул.

Библиографический список включает 35 источников – научные публикации отечественных и зарубежных авторов на русском и английском языках, а также интернет-ресурсы по рассматриваемой теме, на которые в тексте приведены адресные

ссылки, подтверждающие наличие апелляции к оппонентам.

Из резервов улучшения публикации можно отметить следующие. Во-первых, авторам предлагается рассмотреть вариант корректировки названия статьи без предлога «к»: «Исследование и разработка алгоритмов формирования...». Во-вторых, заголовки таблиц лучше разместить в соответствии с принятыми правилами – перед таблицами, а не после них.

Рецензируемый материал соответствует направлению журнала «Программные системы и вычислительные методы», отражает ход и результаты проведенной авторами работы по созданию системы искусственного интеллекта, вызовет интерес у читателей, а поэтому после некоторой доработки в соответствии с высказанными пожеланиями статья рекомендуется к опубликованию.