Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования

«**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**»

(Университет ИТМО)

Факультет программной инженерии и компьютерной техники

Отчёт

По предмету «**Нейротехнологии и аффективные вычисления**»

Тема проекта: «**Определение скуки, вовлеченности, замешательства и фрустрации аудитории в реальном времени с помощью методов искусственного интеллекта**»

Выполнила:

студент группы P33201

Парфенова Ольга Сергеевна

Проверила:

Дата _____отметка _____

Русак А.В._____

г. Санкт-Петербург,
год 2024

Содержание

Введение

В эпоху цифровой трансформации образовательные практики переходят в виртуальное пространство, и онлайн-конференции становятся неотъемлемой частью современного образования. В этом контексте важным вопросом становится не только техническое обеспечение и доступность контента, но и взаимодействие с аудиторией через эмоциональное призму.

Настоящий проект посвящен разработке инновационных методов искусственного интеллекта, направленных на определение эмоционального состояния аудитории образовательных онлайн-конференций. Особый акцент делается на выявлении четырех ключевых эмоций: скуки, вовлеченности, фрустрации и замешательства.

Использование методов искусственного интеллекта в данном контексте предоставляет возможность более глубокого анализа эмоционального отклика слушателей, позволяя адаптировать и улучшать образовательные материалы, методику преподавания и организацию конференций. Это также способствует созданию более эффективных и персонализированных образовательных сценариев, поддерживая активное вовлечение и интерес аудитории.

В данном отчете представлены методы, этапы разработки, а также результаты экспериментов, направленных на создание интеллектуальной системы, способной эффективно определять эмоциональные состояния участников образовательных онлайн-конференций.

1. Анализ предметной области

1.1. Обоснование актуальности темы проекта

В условиях активного внедрения онлайн-образования существует потребность в более глубоком понимании эмоционального состояния аудитории. Эффективность обучения не ограничивается техническими аспектами, и точное определение эмоций, таких как скука, вовлеченность и фрустрация, становится ключевым фактором. Использование методов искусственного интеллекта в этом контексте позволяет не только улучшить качество образовательного процесса, но и создать персонализированные обучающие среды. Такой подход отвечает вызовам современного образования и обеспечивает более эффективное и адаптивное взаимодействие с аудиторией.

1.2. Обзор состояния предметной области

Современная область образовательных технологий и онлайн-обучения стремительно развивается, предоставляя новые возможности для учебного процесса и взаимодействия с аудиторией. Однако, вместе с техническими преимуществами, акцент на эмоциональной составляющей обучения становится все более важным аспектом.
Алгоритмы глубокого обучения используются для определения эмоций в бизнес-продуктах.
Например, MorphCast предоставила SDK для анализа и определения всех видов эмоций, в том числе вовлеченности, внимания и скуки. В том числе компания предоставила плагин для Zoom.
CubicCV выпустила продукт, который может определять и анализировать человеческие эмоции в режиме реального времени, предоставляя вам оперативную информацию об эмоциональном состоянии вашей аудитории.

Zoom IQ анализирует встречу и формирует показатель вовлеченности, с помощью которого можно понять, насколько был заинтересован клиент. Показатель формируется на основе общего времени разговора, длительности пауз между фразами и количества раз, когда клиент перебивает менеджера. Помимо этого, учитывается наличие слов-паразитов во время беседы и эмоциональная окраска фраз.

1.3. Подходы и методы

В последних исследованиях для распознавания вовлеченности используются такие методы, как:

- определение положения головы, глаз
- комбинирование 8 базовых эмоций, таких как счастье, грусть, гнев, страх, отвращение, удивление, презрение и изумление и, на основании них, определение скуки и вовлеченности
- определение мимических выражений

2. Определение цели и задач работы

2.1. Цель

Целью данного проекта является разработка приложения, в реальном времени определяющего уровни четырёх эмоций: скуки, вовлеченности, замешательства и фрустрации, которые выражает лицо человека в кадре при просмотре обучающих материалов. Каждой эмоции присваивается одна из четырех меток, соответствующих уровню интенсивности: очень низкий, низкий, высокий, очень высокий.

2.2. Задачи

1. Разработка модели распознавания эмоций

Создание и обучение модели, способной точно идентифицировать четыре ключевые эмоции (скуку, вовлеченность, замешательство и фрустрацию) по выражению лица пользователя в режиме реального времени.

2. Классификация уровней интенсивности

Реализация механизма, который будет присваивать каждой обнаруженной эмоции одну из четырех меток интенсивности: очень низкий, низкий, высокий, очень высокий, для более детальной оценки эмоциональных состояний пользователя.

3. Оценка эффективности

Оценка точности и производительности разработанной модели.

4. Интеграция в приложение

Разработка приложения, способного в режиме реального времени обрабатывать видеопоток и предоставлять пользователю информацию об уровнях эмоций с соответствующими метками на интерфейсе.

2.3. Аннотация

Страниц 17, приложений 12

Данный проект нацелен на разработку инновационного приложения, способного в реальном времени анализировать эмоциональные состояния пользователя при просмотре обучающих материалов. С использованием передовых методов глубокого обучения, приложение будет точно определять уровни скуки, вовлеченности, замешательства и фрустрации на основе выражения лица.

Ключевые задачи проекта включают в себя создание и обучение модели глубокого обучения для анализа физиономических признаков, интеграцию алгоритмов обработки изображений в реальном времени, и разработку пользовательского интерфейса для визуализации результатов. Итоговым продуктом будет интуитивно понятное и эффективное приложение, способное предоставлять непрерывный анализ эмоционального состояния.

3. Выбор методов и технологий для реализации проекта

3.1. Выбор методов для решения поставленных задач проекта

Для решения поставленных задач проекта по разработке приложения используются следующие методы и технологии:

1. Глубокое обучение (Deep Learning)
Использование сверточных нейронных сетей (CNN) для обучения модели на большом объеме данных с эмоциональными метками. CNN эффективно извлекают признаки из изображений, что важно для анализа выражения лица.

2. Transfer Learning
Использование предобученных моделей, таких как Resnet50 и Xception, обученных на больших наборах данных для обнаружения эмоций (например, на датасетах для распознавания лиц). Это позволяет извлечь полезные признаки изображения, даже при ограниченном объеме данных проекта.

3. OpenCV
Интеграция библиотеки OpenCV для обработки и анализа видеопотока в реальном времени. OpenCV может использоваться для выделения лиц, применения фильтров и других преобразований, необходимых для предобработки данных.

4. Emotion Datasets
Использование обширных датасетов с эмоциональными метками (DAiSEE) для обучения модели. Эти датасеты содержат изображения с различными эмоциональными выражениями, что позволяет модели обучаться на разнообразных данных.

5. Real-Time Image Processing
Применение методов обработки изображений в реальном времени для улучшения производительности и обеспечения моментального анализа эмоций в потоке видео.

6. Многоклассовая классификация
Использование модели для многоклассовой классификации, где каждой эмоции (скука, вовлеченность, замешательство, фрустрация) ставится отдельная классовая метка.

Выбор этих методов позволит создать эффективную и точную модель для анализа эмоций в реальном времени, что важно для достижения целей проекта.

3.2. Выбор и обоснование технологий для реализации проекта

1. Язык программирования Python

Python является одним из наиболее распространенных языков программирования для задач машинного обучения и глубокого обучения. Он обладает обширным сообществом разработчиков, богатым экосистемой библиотек и фреймворков, что упрощает реализацию и поддержку проекта.

2. Библиотека TensorFlow или PyTorch

TensorFlow - это ведущий фреймворк для разработки глубоких нейронных сетей. Предоставляет простой интерфейс для создания и обучения моделей, а также обширные ресурсы для поддержки и обучения.

3. Библиотека Keras

Keras предоставляет высокоуровневый интерфейс для создания и обучения нейронных сетей на основе TensorFlow или PyTorch. Он позволяет легко определить архитектуры моделей и упрощает процесс экспериментирования с различными архитектурами.

4. OpenCV (Open Source Computer Vision Library)

OpenCV предоставляет богатый инструментарий для обработки изображений и видео. Он будет полезен для предобработки данных, выделения лиц и других операций в реальном времени.

5. Transfer Learning с предобученными моделями

Использование предобученных моделей для распознавания эмоций (например, модели, обученные на датасете FER) может улучшить производительность и ускорить процесс обучения, особенно при наличии ограниченного объема данных.

3.3. Описание проекта создаваемого продукта

Основные характеристики продукта:

1. Анализ эмоций

Приложение использует технологии глубокого обучения для точного определения четырех ключевых эмоций – скуки, вовлеченности, замешательства и фрустрации – на основе выражения лица пользователя.

2. Работа в реальном времени

Продукт обеспечивает моментальный анализ эмоционального состояния в реальном времени, что позволяет быстро реагировать на изменения в поведении пользователя.

3. Интуитивный интерфейс

Приложение предоставляет простой и понятный интерфейс для визуализации результатов, позволяя пользователям легко интерпретировать анализ эмоций.

4. Сегментация эмоций

Каждой из четырех эмоций присваиваются уровни интенсивности (очень низкий, низкий, высокий, очень высокий), что обеспечивает более детальное понимание эмоциональных переживаний.

Область применения:

- Образование: Анализ эмоций студентов в реальном времени для оценки эффективности обучения и предоставления персонализированного обучения.

Ожидаемые результаты:

- Эффективность анализа: Разработка точной и стабильной модели анализа эмоций.
- Интуитивный интерфейс: Создание удобного пользовательского интерфейса для взаимодействия с результатами анализа эмоций.

4. Проектирование, разработка, исследование

4.1. Описание продукта

Для определения эмоций используется CNN модель, обученная на датасете.
DAiSEE - это датасет для многоклассовой классификации видео, предназначенный для распознавания эмоциональных состояний пользователей, таких как скука, путаница, вовлеченность и разочарование. Датасет содержит 9,068 видеофрагментов, собранных от 112 пользователей, просматривающих учебные материалы.

Состоит из тренировочной, тестовой и валидационной части.

Из видеофрагментов было извлечено 37568 тренировочных кадров, 12544 тестовых кадров и 10048 валидационных кадров.

Каждому видеофрагменту и, соответственно, кадру из фрагмента соответствует четыре метки, соответствующие уровню интенсивности каждой из четырех эмоций (скука, вовлеченность, замешательство, фрустрация), от 0 до 3, где 0 - очень низкий, 1 - низкий, 2 - высокий, 3 - очень высокий.

Затем обученному классификатору на вход подается изображение с камеры устройства обучающегося, где каждый кадр получает соответствующую метку для каждой из эмоций.

4.2. План экспериментальных исследований

В данном исследовании проводится экспериментальное сравнение предобученных на весах imagenet моделей ResNet50 и Xception.

Описание модели ResNet50

ResNet-50 - это одна из архитектур глубоких нейронных сетей, изначально предназначенная для классификации изображений. Эта модель была представлена в статье "Deep Residual Learning for Image Recognition" (2016) авторами Kaiming He, Xiangyu Zhang, Shaoqing Ren и Jian Sun. ResNet-50 является частью семейства ResNet-моделей, которые стали известными благодаря использованию сверточных блоков с остаточными (residual) связями. Архитектура показала хорошие результаты в задачах классификации эмоций.

Основные характеристики ResNet50:

- Сверточные блоки: Основной строительный блок ResNet-50 - это сверточные блоки с остаточной связью. В отличие от более простых архитектур, ResNet использует остаточные блоки для борьбы с проблемой затухания градиента и улучшения обучения глубоких нейронных сетей.
- Архитектура: ResNet-50 включает в себя 50 слоев, включая сверточные слои, слои глобального среднего пулинга и полносвязанные слои. Он состоит из блоков, в которых используются свертки 1x1, 3x3 и 1x1, а также блоков, содержащих свертки 1x1 и 3x3
- Остаточные связи: Остаточные связи позволяют градиентам легко проходить через блоки нейронов, что облегчает обучение глубоких сетей. Каждый блок добавляет остаточный (residual) вход к выходу блока.
- Глобальное среднее пулинг (Global Average Pooling): В конце сети используется слой глобального среднего пулинга для усреднения признаков по пространственным размерам и создания одномерного вектора, который подается в полносвязанный слой для классификации.
- Веса ImageNet: Исходные веса ResNet-50 часто предварительно обучаются на крупномасштабном наборе данных ImageNet, что позволяет модели выучивать высокоуровневые признаки из изображений.

Описание модели Xception

Xception (Extreme Inception) - это глубокая сверточная нейронная сеть, представленная в статье "Xception: Deep Learning with Depthwise Separable Convolutions" (2017 год) авторства François Chollet. Эта модель представляет собой эволюцию архитектуры Inception и использует сверточные блоки с глубокой разделенной сверткой (depthwise separable convolutions).

Основные характеристики:

- Глубокая архитектура: Xception представляет собой глубокую нейронную сеть, состоящую из 71 слоя. Она строится на идее глубокой сверточной архитектуры с использованием модулей, называемых "глубокие раздельные свертки" (depthwise separable convolutions).
- Глубокие раздельные свертки: Главной особенностью Xception являются глубокие раздельные свертки, которые заменяют традиционные свертки в Inception-модулях.
- Глубокие раздельные свертки разделяют пространственные и глубинные фильтры, что позволяет сети эффективно изучать пространственные и глубинные признаки.
- Блоки идентичности: Как и в архитектуре ResNet, в Xception используются блоки идентичности, чтобы обеспечить эффективное обучение глубоких нейронных сетей. Эти блоки помогают в обработке градиентов и предотвращают проблему затухания градиентов.
- Более высокая точность: В некоторых задачах, таких как классификация изображений, Xception может показывать более высокую точность по сравнению с предыдущими архитектурами, такими как Inception v3.

Ход экспериментального исследования

В ходе экспериментального исследования планируется обучение моделей ResNet50 и Xception на датасете DAiSEE с последующим определением точности на тестовых данных. Точность моделей определяется с помощью sparse categorical accuracy.

Sparse categorical accuracy (разреженная категориальная точность) - это метрика оценки

производительности модели для задач многоклассовой классификации, когда метки классов представлены в виде целых чисел (например, 0, 1, 2 и т.д.), а не в форме one-hot encoding.

Дополнительные действия

- К обоим моделям добавить по два полносвязных слоя, обучить на тренировочных данных и также определить точность полученных моделей
- Fine-tunning модели с наилучшим показателем определения вовлеченности

4.3. Результаты экспериментальных исследований

Обучение моделей производилось на kaggle.
Ссылка на ноутбук:
https://colab.research.google.com/drive/1-MdYcpbRNrBi4WLqogePmhyGEy5HxG6m?usp=sharing

Результат обучения Resnet50 на DAiSEE

Точность определения скуки (y1) - 0.4613
Точность определения вовлеченности (y2) - 0.5061
Точность определения замешательства (y3) - 0.6726
Точность определения фрустрации (y4) - 0.4613

Результат обучения Resnet50 с добавлением полносвязных слоев

Точность определения скуки (y1) - 0.4613
Точность определения вовлеченности (y2) - 0.4563
Точность определения замешательства (y3) - 0.6726
Точность определения фрустрации (y4) - 0.4613
Точность определения вовлеченности снизилась.

Результат обучения Xception на DAiSEE

Точность определения скуки (y1) - 0.4351
Точность определения вовлеченности (y2) - 0.5205
Точность определения замешательства (y3) - 0.6730
Точность определения фрустрации (y4) - 0.4417

Точность определения вовлеченности увеличилась.

Результат обучения Xception с добавлением полносвязных слоев

Точность определения скуки (y1) - 0.3693
Точность определения вовлеченности (y2) - 0.5041
Точность определения замешательства (y3) - 0.6690
Точность определения фрустрации (y4) - 0.3981

Точность классификации сильно снизилась по сравнению с предыдущими моделями.

Точность определения вовлеченности Xception выше по сравнению с ResNet50, что в рамках данного исследования приоритетнее остальных эмоций, поэтому для fine-tunning была выбрана эта модель.

Результат fine-tunning Xception

Точность определения скуки (y1) - 0.4347

Точность определения вовлеченности (y2) - 0.5096

Точность определения замешательства (y3) - 0.6586

Точность определения фрустрации (y4) - 0.4364

Результат fine-tunning Xception с полносвязными слоями

Точность определения скуки (y1) - 0.4118

Точность определения вовлеченности (y2) - 0.5484

Точность определения замешательства(y3) - 0.6720

Точность определения фрустрации (y4) - 0.4291

Как видим точность распознавания вовлеченности выросла до 54%.

Так как у Xception с двумя полносвязными слоями точность определения вовлеченности выше, чем у остальных моделей, для классификации эмоций выбрана именно эта архитектура.

Распознавания эмоций в реальном времени:

```
import cv2
import numpy as np
import tensorflow as tf
from scipy.special import softmax

model = tf.keras.models.load_model('Xception_on_DAiSEE_finetune_fc.h5')

class_labels = {0: "Very Low", 1: "Low", 2:"High", 3: "Very High"}

cap = cv2.VideoCapture(0)

cv2.namedWindow('Engagement Detection', cv2.WINDOW_NORMAL)
cv2.resizeWindow('Engagement Detection', 1000, 800)



while True:
    ret, frame = cap.read()
```

```
    image = cv2.resize(frame, (299, 299))
    image = np.array(image)
    image = np.expand_dims(image, axis=0)

    image = tf.keras.applications.xception.preprocess_input(image)

    predictions = model.predict(image)
    print(predictions)

    probabilities = softmax(predictions, axis=-1)

    boredom_predictions = probabilities[0]
    engagement_predictions = probabilities[1]
    confusion_predictions = probabilities[2]
    frustration_predictions = probabilities[3]

    text_lines = [
    f'Boredom Level: {class_labels[np.argmax(boredom_predictions)]}, Probability:
{round(np.max(boredom_predictions)*100)}%',
    f'Engagement Level: {class_labels[np.argmax(engagement_predictions)]}, Probability:
{round(np.max(engagement_predictions)*100)}%',
    f'Confusion Level: {class_labels[np.argmax(confusion_predictions)]}, Probability:
{round(np.max(confusion_predictions)*100)}%',
    f'Frustration Level: {class_labels[np.argmax(frustration_predictions)]}, Probability:
{round(np.max(frustration_predictions)*100)}%',
    ]

    y_coordinate = 30

    for line in text_lines:
        cv2.putText(frame, line, (10, y_coordinate), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0,
255, 0), 2)
        y_coordinate += 30

    cv2.imshow('Engagement Detection', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

5. Заключение

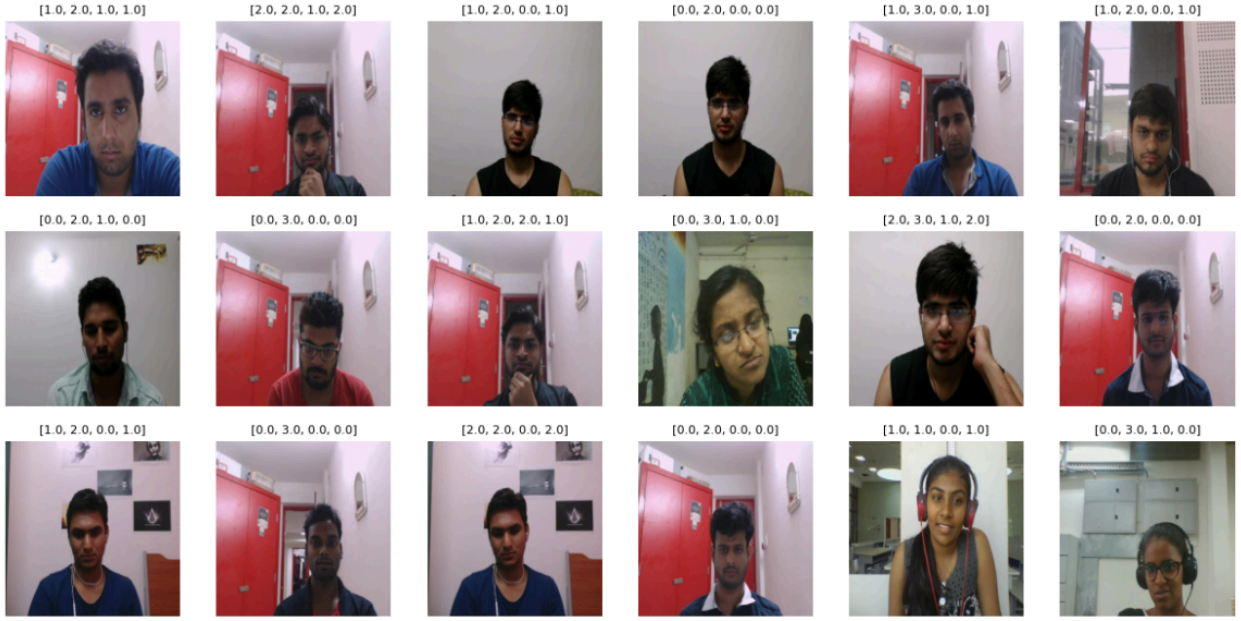
В рамках данного проекта были успешно достигнуты поставленные цели и выполнены задачи, направленные на разработку приложения для реального времени определения уровней четырех ключевых эмоций: скуки, вовлеченности, замешательства и фрустрации.

6. Список использованных источников

1. Prabin Sharma, Shubham Joshi, Subash Gautam, Sneha Maharjan, Salik Ram Khanal, Manuel Cabral Reis, João Barroso, Vítor Manuel de Jesus Filipe. Student Engagement Detection Using Emotion Analysis, Eye Tracking and Head Movement with Machine Learning
2. Ali Abedi, Shehroz S. Khan. Improving state-of-the-art in Detecting Student Engagement with Resnet and TCN Hybrid Network
3. Mayanda Mega Santoni, T. Basaruddin, Kasiyah Junus. Convolutional Neural Network Model based Students' Engagement Detection in Imbalanced DAiSEE Dataset

7. Приложения

Приложение 1. Пример кадров из видеофрагментов DAiSEE и их меток



Приложение 2. Первые несколько слоев ResNet50

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
==================================================================================================
 input_1 (InputLayer)        [(None, 299, 299, 3)]     0         []

 conv1_pad (ZeroPadding2D)   (None, 305, 305, 3)       0         ['input_1[0][0]']

 conv1_conv (Conv2D)         (None, 150, 150, 64)      9472      ['conv1_pad[0][0]']

 conv1_bn (BatchNormalizati  (None, 150, 150, 64)      256       ['conv1_conv[0][0]']
 on)

 conv1_relu (Activation)     (None, 150, 150, 64)      0         ['conv1_bn[0][0]']

 pool1_pad (ZeroPadding2D)   (None, 152, 152, 64)      0         ['conv1_relu[0][0]']

 pool1_pool (MaxPooling2D)   (None, 75, 75, 64)        0         ['pool1_pad[0][0]']

 conv2_block1_1_conv (Conv2  (None, 75, 75, 64)        4160      ['pool1_pool[0][0]']
 D)

 conv2_block1_1_bn (BatchNo  (None, 75, 75, 64)        256       ['conv2_block1_1_conv[0][0]']
 rmalization)

 conv2_block1_1_relu (Activ  (None, 75, 75, 64)        0         ['conv2_block1_1_bn[0][0]']
 ation)

 conv2_block1_2_conv (Conv2  (None, 75, 75, 64)        36928     ['conv2_block1_1_relu[0][0]']
 D)

 conv2_block1_2_bn (BatchNo  (None, 75, 75, 64)        256       ['conv2_block1_2_conv[0][0]']
 rmalization)

 conv2_block1_2_relu (Activ  (None, 75, 75, 64)        0         ['conv2_block1_2_bn[0][0]']
 ation)

 conv2_block1_0_conv (Conv2  (None, 75, 75, 256)       16640     ['pool1_pool[0][0]']
```

Приложение 3. Последние несколько слоев ResNet50

```
conv5_block3_2_bn (BatchNo  (None, 10, 10, 512)         2048      ['conv5_block3_2_conv[0][0]']
rmalization)

conv5_block3_2_relu (Activ  (None, 10, 10, 512)         0         ['conv5_block3_2_bn[0][0]']
ation)

conv5_block3_3_conv (Conv2  (None, 10, 10, 2048)        1050624   ['conv5_block3_2_relu[0][0]']
D)

conv5_block3_3_bn (BatchNo  (None, 10, 10, 2048)        8192      ['conv5_block3_3_conv[0][0]']
rmalization)

conv5_block3_add (Add)      (None, 10, 10, 2048)        0         ['conv5_block2_out[0][0]',
                                                                   'conv5_block3_3_bn[0][0]']

conv5_block3_out (Activati  (None, 10, 10, 2048)        0         ['conv5_block3_add[0][0]']
on)

global_average_pooling2d (  (None, 2048)                0         ['conv5_block3_out[0][0]']
GlobalAveragePooling2D)

y1 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d[0][
                                                                   0]']

y2 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d[0][
                                                                   0]']

y3 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d[0][
                                                                   0]']

y4 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d[0][
                                                                   0]']

==================================================================================================
Total params: 23620496 (90.11 MB)
Trainable params: 32784 (128.06 KB)
Non-trainable params: 23587712 (89.98 MB)
```

Приложение 3. Последние несколько слоев ResNet50 с двумя полносвязными слоями

```
conv5_block3_2_relu (Activ  (None, 10, 10, 512)         0         ['conv5_block3_2_bn[0][0]']
ation)

conv5_block3_3_conv (Conv2  (None, 10, 10, 2048)        1050624   ['conv5_block3_2_relu[0][0]']
D)

conv5_block3_3_bn (BatchNo  (None, 10, 10, 2048)        8192      ['conv5_block3_3_conv[0][0]']
rmalization)

conv5_block3_add (Add)      (None, 10, 10, 2048)        0         ['conv5_block2_out[0][0]',
                                                                   'conv5_block3_3_bn[0][0]']

conv5_block3_out (Activati  (None, 10, 10, 2048)        0         ['conv5_block3_add[0][0]']
on)

global_average_pooling2d (  (None, 2048)                0         ['conv5_block3_out[0][0]']
GlobalAveragePooling2D)

fc1 (Dense)                 (None, 128)                 262272    ['global_average_pooling2d[0][
                                                                   0]']

fc2 (Dense)                 (None, 64)                  8256      ['fc1[0][0]']

y1 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

y2 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

y3 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

y4 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

==================================================================================================
Total params: 23859280 (91.02 MB)
Trainable params: 271568 (1.04 MB)
Non-trainable params: 23587712 (89.98 MB)
__________________________________________________________________________________________________
..
```

Приложение 4. Первые несколько слоев Xception

```
Model: "model_1"
_________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
=================================================================================================
 input_2 (InputLayer)        [(None, 299, 299, 3)]     0         []

 block1_conv1 (Conv2D)       (None, 149, 149, 32)      864       ['input_2[0][0]']

 block1_conv1_bn (BatchNorm  (None, 149, 149, 32)      128       ['block1_conv1[0][0]']
 alization)

 block1_conv1_act (Activati  (None, 149, 149, 32)      0         ['block1_conv1_bn[0][0]']
 on)

 block1_conv2 (Conv2D)       (None, 147, 147, 64)      18432     ['block1_conv1_act[0][0]']

 block1_conv2_bn (BatchNorm  (None, 147, 147, 64)      256       ['block1_conv2[0][0]']
 alization)

 block1_conv2_act (Activati  (None, 147, 147, 64)      0         ['block1_conv2_bn[0][0]']
 on)

 block2_sepconv1 (Separable  (None, 147, 147, 128)     8768      ['block1_conv2_act[0][0]']
 Conv2D)

 block2_sepconv1_bn (BatchN  (None, 147, 147, 128)     512       ['block2_sepconv1[0][0]']
 ormalization)

 block2_sepconv2_act (Activ  (None, 147, 147, 128)     0         ['block2_sepconv1_bn[0][0]']
 ation)

 block2_sepconv2 (Separable  (None, 147, 147, 128)     17536     ['block2_sepconv2_act[0][0]']
 Conv2D)

 block2_sepconv2_bn (BatchN  (None, 147, 147, 128)     512       ['block2_sepconv2[0][0]']
 ormalization)

 conv2d (Conv2D)             (None, 74, 74, 128)       8192      ['block1_conv2_act[0][0]']
```

Приложение 5. Последние несколько слоев Xception

```
block14_sepconv1 (Separabl  (None, 10, 10, 1536)        1582080   ['add_11[0][0]']
eConv2D)

block14_sepconv1_bn (Batch  (None, 10, 10, 1536)        6144      ['block14_sepconv1[0][0]']
Normalization)

block14_sepconv1_act (Acti  (None, 10, 10, 1536)        0         ['block14_sepconv1_bn[0][0]']
vation)

block14_sepconv2 (Separabl  (None, 10, 10, 2048)        3159552   ['block14_sepconv1_act[0][0]']
eConv2D)

block14_sepconv2_bn (Batch  (None, 10, 10, 2048)        8192      ['block14_sepconv2[0][0]']
Normalization)

block14_sepconv2_act (Acti  (None, 10, 10, 2048)        0         ['block14_sepconv2_bn[0][0]']
vation)

global_average_pooling2d_1  (None, 2048)                0         ['block14_sepconv2_act[0][0]']
 (GlobalAveragePooling2D)

y1 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d_1[0
                                                                  ][0]']

y2 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d_1[0
                                                                  ][0]']

y3 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d_1[0
                                                                  ][0]']

y4 (Dense)                  (None, 4)                   8196      ['global_average_pooling2d_1[0
                                                                  ][0]']

==================================================================================================
Total params: 20894264 (79.71 MB)
Trainable params: 32784 (128.06 KB)
Non-trainable params: 20861480 (79.58 MB)
__________________________________________________________________________________________________
```

Приложение 6. Последние несколько слоев Xception с двумя полносвязными слоями

```
block14_sepconv1_bn (Batch  (None, 10, 10, 1536)        6144      ['block14_sepconv1[0][0]']
Normalization)

block14_sepconv1_act (Acti  (None, 10, 10, 1536)        0         ['block14_sepconv1_bn[0][0]']
vation)

block14_sepconv2 (Separabl  (None, 10, 10, 2048)        3159552   ['block14_sepconv1_act[0][0]']
eConv2D)

block14_sepconv2_bn (Batch  (None, 10, 10, 2048)        8192      ['block14_sepconv2[0][0]']
Normalization)

block14_sepconv2_act (Acti  (None, 10, 10, 2048)        0         ['block14_sepconv2_bn[0][0]']
vation)

global_average_pooling2d_2  (None, 2048)                0         ['block14_sepconv2_act[0][0]']
 (GlobalAveragePooling2D)

fc1 (Dense)                 (None, 128)                 262272    ['global_average_pooling2d_2[0
                                                                  ][0]']

fc2 (Dense)                 (None, 64)                  8256      ['fc1[0][0]']

y1 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

y2 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

y3 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

y4 (Dense)                  (None, 4)                   260       ['fc2[0][0]']

==================================================================================================
Total params: 21133048 (80.62 MB)
Trainable params: 271568 (1.04 MB)
Non-trainable params: 20861480 (79.58 MB)
__________________________________________________________________________________________________
```

Приложение 7. Результат обучения Xception на DAiSEE

```
xception_model = load_model(f"/kaggle/working/model/Xception_on_DAiSEE.h5")
xception_accuracy = xception_model.evaluate(test_set)
```

```
196/196 [==============================] - 2816s 14s/step - loss: 4.2366 - y1_loss: 1.2017 - y2_loss: 0.9246 - y3_loss: 0.9306 -
y4_loss: 1.1796 - y1_sparse_categorical_accuracy: 0.4351 - y2_sparse_categorical_accuracy: 0.5205 - y3_sparse_categorical_accura
cy: 0.6730 - y4_sparse_categorical_accuracy: 0.4417
```

Приложение 8. Результат обучения Xception с добавлением двух полносвязных слоев

```
xception_model = load_model(f"/kaggle/working/model/Xception_on_DAiSEE_fc.h5")
xception_accuracy = xception_model.evaluate(test_set)
```

```
196/196 [==============================] - 2905s 15s/step - loss: 4.0495 - y1_loss: 1.1778 - y2_loss: 0.8632 - y3_loss: 0.8644 -
y4_loss: 1.1441 - y1_sparse_categorical_accuracy: 0.3693 - y2_sparse_categorical_accuracy: 0.5041 - y3_sparse_categorical_accura
cy: 0.6690 - y4_sparse_categorical_accuracy: 0.3981
```

Приложение 9. Результат обучения ResNet50 на DAiSEE

```
196/196 [==============================] - 2421s 12s/step - loss: 4.0060 - y1_loss: 1.1260 - y2_loss: 0.8651 - y3_loss: 0.8892 -
y4_loss: 1.1257 - y1_sparse_categorical_accuracy: 0.4613 - y2_sparse_categorical_accuracy: 0.5061 - y3_sparse_categorical_accura
cy: 0.6726 - y4_sparse_categorical_accuracy: 0.4613
```

Приложение 10. Результат обучения ResNet50 с добавлением двух полносвязных слоев

```
196/196 [==============================] - 2631s 13s/step - loss: 4.0048 - y1_loss: 1.1218 - y2_loss: 0.8885 - y3_loss: 0.8690 -
y4_loss: 1.1257 - y1_sparse_categorical_accuracy: 0.4613 - y2_sparse_categorical_accuracy: 0.4563 - y3_sparse_categorical_accura
cy: 0.6726 - y4_sparse_categorical_accuracy: 0.4613
```

Приложение 11. Результат fine-tunning Xception

```
xception_model = load_model(f"/kaggle/working/model/Xception_on_DAiSEE_finetune.h5")
xception_accuracy = xception_model.evaluate(test_set)
```

```
196/196 [==============================] - 2726s 14s/step - loss: 4.0228 - y1_loss: 1.1414 - y2_loss: 0.8625 - y3_loss: 0.8668 -
y4_loss: 1.1522 - y1_sparse_categorical_accuracy: 0.4347 - y2_sparse_categorical_accuracy: 0.5096 - y3_sparse_categorical_accura
cy: 0.6586 - y4_sparse_categorical_accuracy: 0.4364
```

Приложение 12. Результат fine-tunning Xception с двумя полносвязными слоями

```
xception_model = load_model(f"/kaggle/working/model/Xception_on_DAiSEE_finetune_fc.h5")
xception_accuracy = xception_model.evaluate(test_set)
```

```
196/196 [==============================] - 2705s 14s/step - loss: 3.9492 - y1_loss: 1.1308 - y2_loss: 0.8425 - y3_loss: 0.8539 -
y4_loss: 1.1219 - y1_sparse_categorical_accuracy: 0.4118 - y2_sparse_categorical_accuracy: 0.5484 - y3_sparse_categorical_accura
cy: 0.6720 - y4_sparse_categorical_accuracy: 0.4291
```