University of Southampton

Faculty of Engineering and Physical Sciences

Electronics and Computer Science

Multi-task Learning for Student Mental State Recognition from Facial Expressions in-the-wild

by

Reshma Abraham

8^{th} September, 2022

Supervisor: Dr. Indu P. Bodala
Second Examiner: Dr. Mohammad D. Soorati

A dissertation submitted in partial fulfilment of the degree
of MSc. Artificial Intelligence

UNIVERSITY OF SOUTHAMPTON
FACULTY OF ENGINEERING AND PHYSICAL SCIENCES
ELECTRONICS AND COMPUTER SCIENCE

ABSTRACT

Multi-task Learning for Student Mental State Recognition from Facial Expressions in-the-wild

by Reshma Abraham

E-Learning platforms are most effective when personalised, leading to higher engagement levels and promoting positive, sustained use. In their current state, these platforms have been designed as a one-size-fits-all solution with no adaptation to the emotional and cognitive abilities of students. By understanding learner engagement and other mental states at various junctures of the learning process and how these change throughout the learning activity, we can intuitively design interfaces that support better learner cognition, help decrease dropout rates, and provide a personalised learning experience.

This paper proposes three novel Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) hybrid networks for multi-task learning of learner mental state recognition from facial expressions in the wild. The three networks differ in their feature extraction architectures: Feature Extractor 1 is based on FaceNet, Feature Extractor 2 is based on ResNet18, and Feature Extractor 3 is based on Inception-ResNet-V1. Student mental state recognition in the wild has been formulated as a Spatio-temporal problem, and the proposed networks are trained and evaluated on the DAiSEE e-learning video dataset. The deep CNN models serve as feature extractors to produce spatial feature embeddings from successive video frames. The LSTM-based sequence learning arm captures the temporal changes in videos to determine how the mental states change over time. The project also investigates the effects of modifying various parameters and specifications of these networks. The results of the project successfully demonstrated the potential of these new approaches in e-learning mental state recognition.

Acknowledgements

Firstly, I would like to thank my supervisor, Dr Indu P. Bodala, for letting me explore this topic and providing crucial guidance throughout its development. I'm equally grateful to my second examiner, Dr Mohammad D. Soorati, whose feedback helped me strengthen my results and discussion chapters.

Thanks to the staff, lecturers and course supervisors at the University of Southampton for expertly organising and delivering the MSc in Artificial Intelligence course and providing the conceptual knowledge needed for this project. A special mention to the HPC support team for their quick responses to all my software queries.

Lastly, I am eternally grateful to my friends and family for their constant support throughout my master's.

Dedicated to Shajy George, my ever-supportive Dad, the man who taught me to never limit my dreams....

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

You must change the statements in the boxes if you do not agree with them.

We expect you to acknowledge all sources of information (e.g. ideas, algorithms, data) using citations. You must also put quotation marks around any sections of text that you have copied without paraphrasing. If any figures or tables have been taken or modified from another source, you must explain this in the caption and cite the original source.

I have acknowledged all sources, and identified any content taken from elsewhere.

If you have used any code (e.g. open-source code), reference designs, or similar resources that have been produced by anyone else, you must list them in the box below. In the report, you must explain what was used and how it relates to the work you have done.

I have not used any resources produced by anyone else.

You can consult with module teaching staff/demonstrators, but you should not show anyone else your work (this includes uploading your work to publicly-accessible repositories e.g. Github, unless expressly permitted by the module leader), or help them to do theirs. For individual assignments, we expect you to work on your own. For group assignments, we expect that you work only with your allocated group. You must get permission in writing from the module teaching staff before you seek outside assistance, e.g. a proofreading service, and declare it here.

I did all the work myself, or with my allocated group, and have not helped anyone else.

We expect that you have not fabricated, modified or distorted any data, evidence, references, experimental results, or other material used or presented in the report. You must clearly describe your experiments and how the results were obtained, and include all data, source code and/or designs (either in the report, or submitted as a separate file) so that your results could be reproduced.

The material in the report is genuine, and I have included all my data/code/designs.

We expect that you have not previously submitted any part of this work for another assessment. You must get permission in writing from the module teaching staff before re-using any of your previously submitted work for this assessment.

I have not submitted any part of this work for another assessment.

If your work involved research/studies (including surveys) on human participants, their cells or data, or on animals, you must have been granted ethical approval before the work was carried out, and any experiments must have followed these requirements. You must give details of this in the report, and list the ethical approval reference number(s) in the box below.

My work did not involve human participants, their cells or data, or animals.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

There has been an accelerated growth in e-learning and online courses in the past few years. Digital learning technologies provide novel ways for students to learn and instructors to teach, anywhere and at any time. Students have the flexibility to choose between synchronous or asynchronous learning with these technologies. However, e-learning platforms are most often designed as a one-size fits all solution, making them boring, impersonal and non-interactive in the long run. Additionally, learning in a technology-rich setting requires students to apply diverse skills and self-directed learning strategies to successfully assimilate learning content. Different levels of skill development and various pragmatic constraints can cause some students to experience various cognitive challenges and feelings of frustration, boredom, or confusion during the learning process. These behaviours cannot be easily observed in digital settings by instructors. Even if the students decide to communicate the challenges they face when learning with technology, the communication is often not real-time - it's either through forms or emails. Due to the lag in communication, it is hard for instructors to understand when, for instance, confusion arose during the learning activity and for how long. The main drawback of existing e-learning systems is that they lack real-time interactive feedback to users during the content delivery process and do not account for the context-sensitive nature of learning in comparison to traditional classroom learning, resulting in lower learning outcomes. Learning technologies rarely provide students with appropriate interventions or feedback mechanisms to address their cognitive and affective struggles.

Student engagement and emotional states are tightly correlated to learning gains as emotions influence attention and attention drives learning. Technology is most effective when personalised as it leads to higher levels of engagement and promotes positive, sustained use. This holds true for e-learning systems too and has been highlighted by [Bartolomé et al. (2018)] in their study. Thus, the benefits from e-learning rely

heavily on the extent to which they take into account student affect and cognition. By understanding learner engagement and other cognitive states at various junctures of the learning process and how these intertwine and span throughout the learning activity, we can intuitively design interfaces that support better learner cognition, help decrease dropout rates [Ramesh et al. (2013)], and provide a personalised learning experience.

1.2 Background and Objective

Affective Computing is a field of computer science that allows digital systems like computers and robots to recognise, quantify and respond to human emotions and cognitive states [Picard (2000)]. It focuses on evaluating emotions expressed across multiple modalities while also considering the context of an interaction. Emotions play a crucial role in human interactions, communications, decision making, perception, memory, and creativity. Thus, it only makes sense to integrate the detection of affective and cognitive states into learning environments, which can subsequently be used to identify appropriate real-time feedbacks, responses and learning strategies. Additionally, such emotional cues can be used to identify parts of a lecture or course that confuses students or are least engaging.

Affect, in psychology, refers to the underlying experience of feeling, emotion or mood [wik (2022)]. Identifying and interpreting human emotions require the knowledge of the stimuli that evoked them. Therefore, the context of the emotion is crucial in deciding the appropriate emotion classification for a problem. For an e-learning context, basic emotional states like happiness, sadness, etc.. have to be upgraded to mental states like engagement, boredom, motivation, attention and so on, to also accommodate information about student cognition.

There has been an upswing in work around student affect modelling [Conati and Maclaren (2009)], the detection of learner frustration and stress [Burleson (2006), McQuiggan et al. (2007)], the modelling of learners' emotional states [Vicente and Pain (2002)], and the assessment of student self-efficacy [Carole and Hyokyeong (2005)]. By incorporating such work into learning environments, these platforms can be tailored to each user's need. Research has shown that adaptive e-learning improves the effectiveness of e-learning systems [Shute and Towle (2018)]. A growing number of robotics researchers are also exploring affective computing approaches for pedagogy agents and have reported improvements in student performance [Elliott et al. (1999)]. Within the Intelligent Tutoring Systems (ITS) community, several research have shown that students who interacted with robots that demonstrated any of the three types of personalisation - nonverbal behavior, verbal behavior, and adaptive content progression, showed increased learning gains and sustained engagement [Ammar et al. (2010), Baxter et al. (2017)].

I identified state-of-the-art (SOTA) models in affective computing from the results of the **Emotion Recognition in the Wild (EmotiW)** challenges, which have several sub-challenges, two of which are "Engagement Prediction in the Wild (EW)" and "Audio-video based Group Emotion Recognition (AV)". I realised limited models from affective computing have been explored within the e-learning domain through my lterature review. Therefore, the overarching objective of this work is to "**Apply a few SOTA architectures in affective computing to the e-learning context and Investigate if they perform well in detecting learner mental states**." In other words, **bridge the gap between basic emotion recognition in affective computing and mental state recognition in e-learning**.

1.3 Contributions of this Work

In this work, I propose an end-to-end pipeline for "**Multi-task Learning for Student Mental State Recognition from Facial Expressions in-the-wild**". The contribution of this work is three-fold:

1. **A Proof-of-Concept(POC) of multi-task video classification of student mental states**: Most existing studies in e-learning either attempt to hard classify student mental states [Bahreini et al. (2016); Sabourin et al. (2011); Zhang and Gu (2020)] or work only on the learner engagement state [Thomas and Jayagopi (2017); Krithika and GG (2016); Khenkar and Jarraya (2022)]. This means that most models do not allow multiple states to occur simultaneously. For instance, a student could be highly engaged in the course but confused in a task. In such a case, it is more beneficial to quantify the level of engagement with confusion to provide an appropriate feedback. This is where a multi-task learning of mental state classification becomes essential. Therefore, this work presents a POC of multi-task learning for video classifications trained and evaluated on the DAiSEE e-learning video dataset [Gupta et al. (2016)]. The proposed approach predicts the level (very low, low, high and very high) of four student mental states - Engagement, Confusion, Boredom, Frustration.

2. **Spatio-Temporal analysis of features for Affect Recognition**: Facial emotion recognition (FER) in the wild is a static problem as it only accounts for spatial features. In e-learning, studies by [Upadhyay et al. (2021); Thomas and Jayagopi (2017)] have performed frame-level learner emotion recognition. However, when FER is extended to e-learning, it has to be formulated as a dynamic problem, since now changes in the student mental states over time have to be taken into account. In this work, I have combined deep Convolutional Neural Network (CNNs) based feature extractors and sequence learning via LSTMs, to capture the spatio-temporal dynamics of student emotions.

3. **Performance investigation of three state-of-the-art networks as feature extractors for online learning problems**: I identified three potential feature extractor architectures from the existing literature and analysed their performance against each other and the baseline models mentioned in the DAiSEE paper [Gupta et al. (2016)]. The three architectures explored are:

   - **Feature Extractor 1**: Inspired by a teacher network proposed by [Schoneveld et al. (2021)] trained on Google FEC dataset [Vemulapalli and Agarwala (2019)] and AffectNet dataset [Mollahosseini et al. (2017)]
   - **Feature Extractor 2**: ResNet18 [He et al. (2016)] pre-trained on ImageNet dataset [Deng et al. (2009)]
   - **Feature Extractor 3**: Inception-ResNet-V1 [Szegedy et al. (2015)] pre-trained on VGGFace2 [Cao et al. (2018)]

   Note that the above mentioned models have not been either utilised as feature extractors for e-learning or paired with LSTMs for Student Affective recognition before.

1.4 Outline

Chapter 2 offers a literature review of the research in the field and some general insights into any theoretical knowledge needed to understand this project. Chapter 3 highlights the methodology and approach taken to conduct the research, along with details of the datasets and the models used. In chapter 4, the results of the experiments conducted have been presented, together with its analysis. Finally, chapter 5 comprises the conclusion, further reflection on the work, and suggestions for future work and research directions.

Chapter 2

Literature Review

This chapter intends to lay a conceptual foundation to the work undertaken. Section 2.4 also highlights some inspirations of my project.

2.1 Emotion Modelling and Affect Representations

Affect, in psychology, refers to the underlying experience of feeling, emotion or mood [wik (2022)]. Affect is subjective, and in the existing literature, affective facial behaviors are quantified using three types of models/representations: **Categorical representation** where affect is represented as a set of discrete states, **Facial Action Coding System (FACS) model** where facial actions are described in terms of Action Units (AUs) [Ekman and Friesen (1978)] and **Dimensional representation** [Schoneveld et al. (2021)] which is suitable for distinguishing subtly different displays of affect and encoding small changes in the intensity of emotions on a continuous scale. The most popular categorisation of emotions is the seven basic classifications: Anger, Disgust, Fear, Happiness, Sadness, Surprise, and Neutral [Mollahosseini et al. (2017)]. The FACS model proposed by Paul Ekman [Ekman and Friesen (1978)] is about the movement of facial muscles described by 46 Action Units. It is a system of mapping facial muscles to the descriptors of those muscles and this can be achieved through methods like EMFACS [Renneberg et al. (2005)]. The Valence and Arousal Space (VA-Space) is the most common dimensional emotion representation. Emotional valence determines whether an emotion is positive or negative, whilst arousal determines whether the affect is passive or active.

2.2 Datasets

Emotions captured for affective computing datasets can be acted/posed, spontaneous and natural or induced. In the early days of affective computing, data collection was conducted in controlled lab settings where subjects acted out basic emotions like anger. In [Kaliouby and Robinson (2005)], professional actors acted emotions and had their expressions recorded. Acted/posed data can be exaggerated; thus, emotion detection models trained on acted data don't work well in naturalistic settings [Barrett et al. (2019)]. Therefore, most recent studies have focused on collecting data using emotional cues or situational procedures to elicit emotions under naturalistic settings [Siedlecka and Denson (2019)]. Emotional cues include visual stimuli, sound, music and films. Examples of situational methods include giving gifts, anticipating failure in easy tasks, playing computer games [Sabourin et al. (2011)], completing driving tasks [Healey and Picard (2005)], speaking tasks [Siedlecka and Denson (2019)], taking mathematical tests, and taking a course. Virtual reality has also been recently explored to elicit emotions through immersive environments [Marín-Morales et al. (2020)]. Emotions can be detected from several modes of human communication, such as speech, facial expressions, gestures, and poses, with vocal and facial expressions being the most popular. However, the main caveat of these modes is that cultural backgrounds, gender, and age can influence them. As a result, a few researchers have been investigating emotion recognition based on physiological signals, which can be monitored by wearable sensors [Shu et al. (2018)]. Figure 2.1 shows the different types of existing emotion recognition datasets.

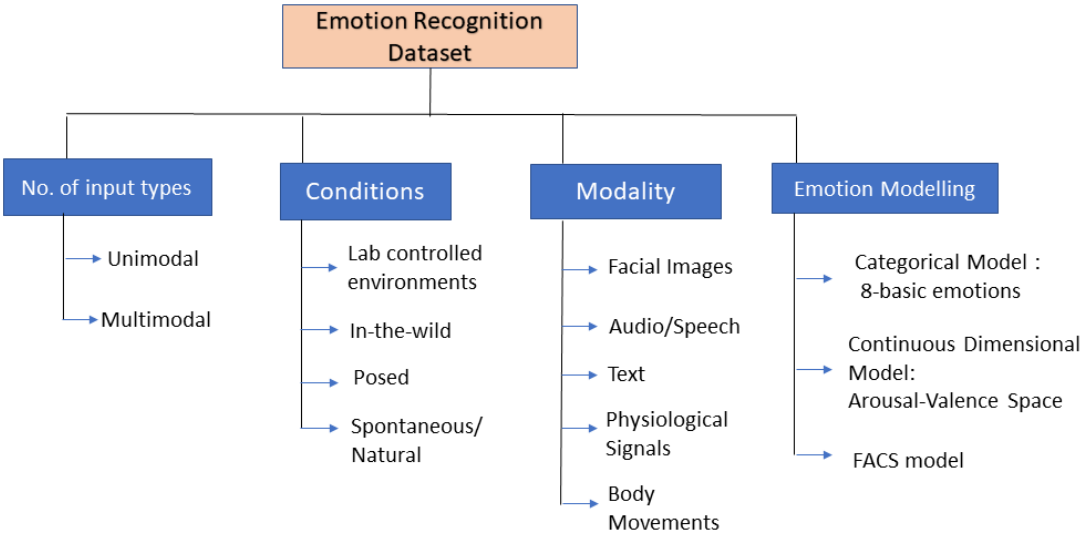


FIGURE 2.1: Taxonomy of Emotion Recognition Datasets

2.2.1 Popular Emotion Recognition Datasets

2.2.1.1 AffectNet

The AffectNet dataset [Mollahosseini et al. (2017)] contains 1,000,000 images queried from the internet using 1250 emotion-related tags in six different languages corresponding to gender, age, and ethnicity. It is currently the largest dataset of categorical and dimensional models of affect in the wild. The dataset has labels for the six basic expressions, and it contains a "None" type that is used for emotional states like bored, tired, confused, focused etc. The annotations are focused on the affect type rather than intensity. The images are also annotated for valence and arousal in the continuous domain, and the dataset contains 450,000 subjects. Figure 2.2 shows sample images from AffectNet on the valence and arousal space.

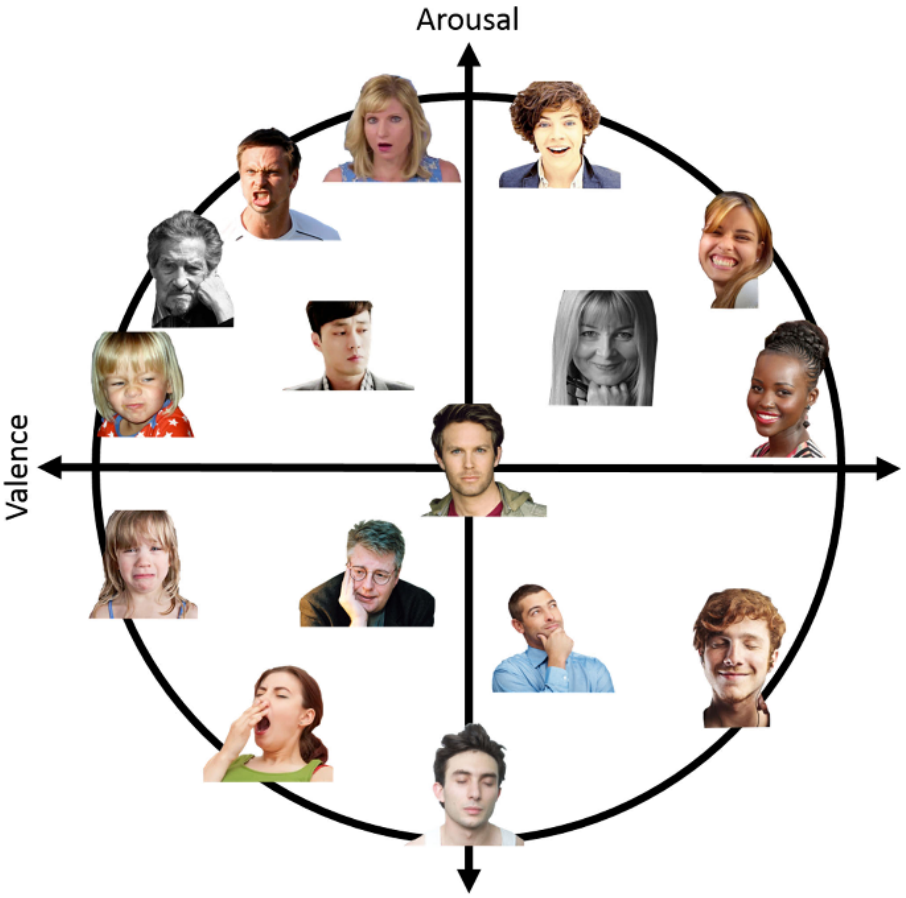


FIGURE 2.2: Sample images from AffectNet on the valence and arousal space. (Source: Mollahosseini et al. (2017))

2.2.1.2 Aff-Wild

The Aff-Wild Database [Zafeiriou et al. (2017)] is the largest dataset of in-the-wild videos collected from YouTube that measures continuous affect in the valence-arousal space. These videos contain the spontaneous emotions of participants while watching a particular video, performing an activity, or reacting to a practical joke. Three human raters have annotated frame-by-frame videos, utilizing a joystick-based tool to rate valence and arousal. Though Aff-Wild provides great dimensional modelling in the wild, it has a small subject variance with only 500 subjects.

2.2.1.3 RECOLA

The REmote COllaborative and Affective Interactions (RECOLA) dataset [Ringeval et al. (2013)] contains videos of spontaneous interactions of 46 participants collected during a virtual discussion. Multimodal data, including audio, video, ECG, and EDA, were recorded continuously and synchronously during the first five minutes of an interaction. Annotators continuously measured the emotional state of all sequences on the valence and arousal dimensions before and after the task using the Self-Assessment Manikin (SAM) [Bradley and Lang (1994)]. While RECOLA is a good multimodal dataset in the dimensional space, it contains only 46 subjects monitored in lab controlled settings.

2.2.1.4 FER-Wild

The Facial Expression Recognition (FER) dataset [Goodfellow et al. (2013)] was created using the Google Search API that matched a set of 184 emotion-related keywords, resulting in a dataset containing 35,887 images of the seven basic expressions captured in-the-wild. Due to the resolution and quality of the FER images, facial landmark detectors have trouble extracting landmarks, and only categorical models are provided in this dataset.

2.2.2 Learner State Recognition Datasets

In the area of learner engagement detection, most research use their own in-house datasets and very few are made publicly available online. Various modalities have been utilized to automatically detect students' engagement, including students' facial and body images [Khenkar and Jarraya (2022)], videos [Gupta et al. (2016), Delgado et al. (2021)], audio [Dhall et al. (2018)] and Electrocardiogram (ECG) signals [Ohn et al. (2020)].

2.2.2.1 DAiSEE

The DAiSEE dataset [Gupta et al. (2016)] captures learner engagement in online learning settings. It includes videos of 112 individuals, where 80 males and 32 females are in uncontrolled environments, such as dorm rooms, labs and libraries, under three different illumination settings captured with a webcam while the learner watches some video tutorial. There are four annotations for four mental state labels—engaged, bored, confused, and frustrated, relying on the "wisdom-of-the crowd". In addition, annotations are rated from 0 to 3 according to their intensity. Such an annotation has the advantage of providing information about the degree/intensity of a specific state.

2.3 Emotion Recognition in the Wild

Research on affect recognition has seen a considerable shift from the study of laboratory-controlled databases to real-world databases collected in the wild [Schoneveld et al. (2021)]. In a controlled environment, specific emotions are evoked, and subjects are made to portray a particular emotion, disregarding external factors that could distract or influence their emotional states. Participants are made to assume a static position to limit face, pose and gesture occlusions, illumination variations, noise, etc. **In-the-wild** studies are those where experimenters don't completely control subjects' emotion elicitation process; real-life events evoke emotions. In these studies, subjects are continuously monitored while they work on everyday tasks or engage in domain-specific activities - for instance, spectators' emotions are captured while watching a film or advertisement in media analytics [Muszynski et al. (2018)]. Today, automatic emotion recognition of acted/posed visual and audio expressions is possible with high accuracy but since spontaneously occurring behaviour varies more widely in its audio profile, visual aspects, and timing, in-the-wild emotion recognition poses a more significant challenge.

2.4 EmotiW Challenge Submissions

This section discusses about three publications from the Emotion Recognition in the Wild (EmotiW) challenges that achieved exemplary performance and were commended for their innovative approaches. Following are my key findings and a summary of what I have learnt from these models for my project:

- [Vielzeuf et al. (2018)] was ranked 4^{th} in the EmotiW2018 challenge [Dhall et al. (2018)] and was noted for being lightweight. The authors used ResNet18 [He et al. (2016)] as their feature extractor and achieved a state-of-the-art accuracy of 60.64% on the test set of Acted Facial Expression in Wild (AFEW) videos dataset. They

stated that the small amount of training clips (773) made it hard for the model to learn, prompting them to pre-train the model exclusively on AffectNet facial emotions database. Their model was trained on an 8-emotion classification task and Arousal-Valence prediction task. They used temporal pooling, which they argue performed slightly better than their LSTM model for sequence learning. They also proposed a simple frame selection mechanism to weight the images of a sequence.

- [Liu et al. (2018)] used multiple CNN models and fused the results in a way that placed them 1^{st} in the EmotiW2017 challenge with 61.87% accuracy. They used CNN models: DenseNet [Huang et al. (2017)] and Inception-ResNet-V1 [Szegedy et al. (2015)] as feature extractors and performed frame-level classification. For temporal learning, they used LSTM with 128 hidden nodes fed with sequences with 16 frames of size (224x224x3) and performed significant data augmentation for training. They too reported insufficient training video clips deteriorated their model performance initially, which they overcame by collecting their own large dataset of emotion recognition video clips called STED.

- [Knyazev et al. (2017)] ranked 2^{nd} in the EmotiW2017 challenge with 60.03% accuracy. For feature extraction from frames, the authors used VGGFace [Cao et al. (2018)] and three proprietary state of the art face recognition networks which they have referred as FR-Net-A, FR-Net-B and FR-Net-C. Through their results they have highlighted the importance of fine-tuning pre-trained models for FER problems. By fine-tuning each of these models on the AFEW dataset, they were able to achieve 10% improvement in accuracy with VGGFace model and around 15% accuracy improvement with the other three networks. They also noted that with LSTMs the accuracy of classification improved.

My conclusions from the above papers:

1. **Importance of training dataset size**: The lack of training data in the challenge impeded the models from performing well, which the authors resolved through data augmentation or by pre-training the models on similar datasets.

2. **Class Imbalance**: Class imbalance causes serious deterioration in the performance and has been taken care through class-wise weighting, loss penalisation and data augmentation.

3. **Transfer Learning with Sequence learning** These papers through their results demonstrate the success of combining transfer learning with sequence learning via LSTMs for spatio-temporal analysis. The authors have used Convolutional Neural Network (CNNs) for feature extraction, which are then directly fed into a Reccurrent Neural Network (RNNs) for classification.

4. **Time and Storage**: As far as implementation is concerned, these submissions seem to have greater computational capacities and more time to experiment / train than what is available for my project, so my paper is unlikely to outperform in terms of results. However, my goal is to convince readers through my experiments that, given more time and computational resources, the networks proposed in this work can outperform the baseline models in e-learning problems.

2.5 Underlying concepts and Related work

2.5.1 Multi-task Learning

Multi-task learning is a machine learning approach in which the model attempts to learn multiple tasks simultaneously, optimising multiple loss functions at once. [Xia and Liu (2015)] applied multi-task learning to leverage arousal and valence information for acoustic emotion recognition based on the deep belief network (DBN) framework. They attempted to predict the arousal-valence values of each category of emotions. [Zhang and Gu (2020)] presented a Multi-Task Affect Net(MTANet) model at the Affective Behavior Analysis in-the-Wild Challenge. This multi-task network based on SE-ResNet modules estimated and recognised three affective representations: valence and arousal, action units, and seven basic emotions simultaneously. In my work, there are four tasks, each one predicts the levels/intensity of one of the four student mental states - Engagement, Confusion, Boredom and Frustration.

2.5.2 Transfer Learning

The goal of transfer learning is to apply a neural network trained on one problem to solve a similar one by converting input features into vector embeddings. These pre-trained models can also be trained from scratch. This method allows us to save time and computational costs, while also making up for insufficient training data. [Upadhyay et al. (2021)] employed transfer learning for engagement detection on the DAiSEE dataset. They proposed a custom Xception Model [Chollet (2017)] pre-trained on billions of images for multi-task classification. The proposed method slightly outperformed the frame-level classification benchmarked by DAiSEE's baseline models.

2.5.3 Sequence Learning

Sequential Learning is a technique to train machine learning models to learn from input data sequences like text streams, audio clips, video clips and time-series data. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTMs) are well-known methods in sequence modelling. [Liao et al. (2021)] proposed Deep Facial Spatio-Temporal

Network (DFSTN) for students' engagement detection where they use two modules, a pretrained SE-ResNet-50 for extracting spatial features from faces, and an LSTM with global attention for generating an attentional hidden state. They evaluated their method on the DAiSEE dataset and achieved an accuracy of 58.84%. [Schoneveld et al. (2021)] coupled LSTMs with feature extractors trained through self-knowledge distillation to predict arousal-valence values of RECOLA remote video dataset to achieve an accuracy of 71%, outperforming the baselines. My Feature Extractor1 largely draws inspiration from this paper.

2.5.4 Feature representation

Spatial features are vector embeddings that contain locations or spatial information related to an individual images or datapoints. Temporal features refer to any feature that is associated with or changes over time like video clips or sequences of audio. RNNs are capable of capturing the temporal nature of such sequences. [Abedi and Khan (2021)] approached student engagement detection as a spatio-temporal problem with Resnet and TCN Hybrid Network. ResNet18 [He et al. (2016)] was used to extract facial feature embeddings while TCN was used to observe the temporal changes in emotions. My Feature Extractor2 draws inspiration from this work.

2.5.5 Other related studies

[Sabourin et al. (2011)] developed a Bayesian network to predict student interactions during a game based learning task on Crystal Island environment. They were able to highlight the importance of temporal information in predicting learner engagement. [Khenkar and Jarraya (2022)] analysed spatio-temporal features from student micro body gestures and mapped it to student engagement levels. They employed transfer learning using C3D model [Tran et al. (2015)] trained on Sports-1M dataset [Karpathy et al. (2014)]. Although they achieved an accuracy of 94% on their in-house dataset, the limited number of participants and variance in the dataset collected leaves readers skeptical of its performance.

Chapter 3

Methodology

3.1 Inspiration

Most studies focusing on learner affect rely on only one modality, namely images, since video cameras and webcams can be easily deployed in e-learning environments as low-cost, ubiquitous, and unobtrusive means of detecting learner affect. Therefore, this research problem was initially formulated to assess student engagement using multimodal data, including facial expressions, body movements and poses, eye gaze direction, head orientation and screen interactions captured while a student attempts an online course. It was thus imperative to acquire a multimodal dataset for learner affect recognition.

Conducting a good data collection experiment and study had several constraints given the duration of the dissertation. For reliable results, the study would need around 10 participants, with an equal proportion of women and men, all screened and assessed as suitable for the study. Additionally, the experiment would require reliable self-reporting tools, questionnaires, multimodal fusion procedures, synchronisation processes and, most importantly, a good data annotation scheme. Due to these reasons, data collection proved infeasible.

A multimodal dataset most closely related to the problem statement was REmote COllaborative and Affective Interactions (RECOLA) dataset [Ringeval et al. (2013)]. In RECOLA, the participants' spontaneous interactions were recorded during a virtual discussion that manipulated their moods. It contains 27 audio-visual recordings of 5 minutes of interactions. Unfortunately, the authors were unresponsive to the data access requests.

So, I decided to direct my research to novel techniques that can outperform the state-of-the-art results on **multi-level learner affective state classification in-the-wild**. DAiSEE dataset [Gupta et al. (2016)] appeared an excellent choice for this work due to the following reasons:

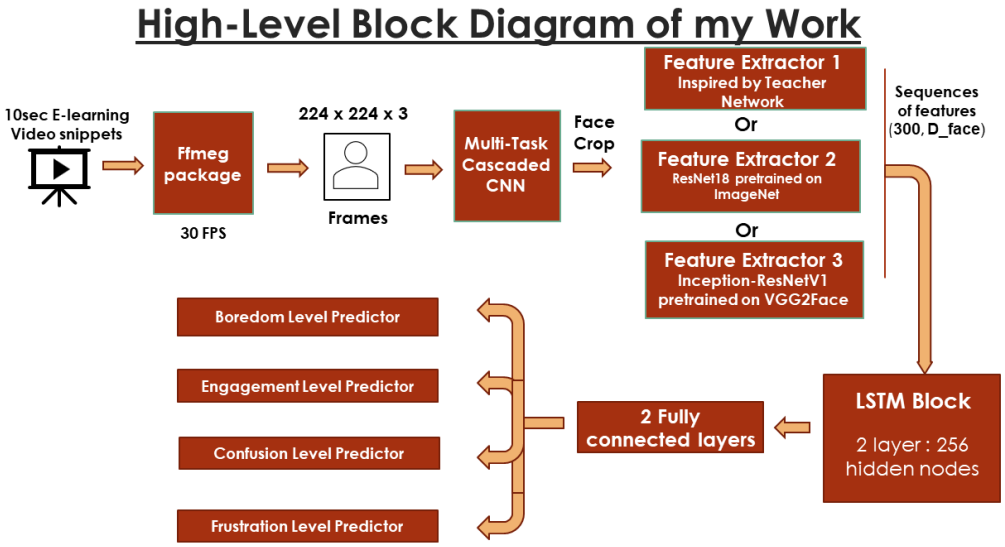


FIGURE 3.1: Methodology Overview

1. It is the first multi-label video classification dataset comprising 9068 video snippets captured from 112 users for recognizing learner affective states of boredom, confusion, engagement, and frustration in-the-wild. The dataset has four levels of labels for each affective state - very low, low, high, and very high. Such labelling allowed the development of a multitask learning model, indicating a student's affective state as well as the degree of it. The model could be trained to make soft decisions regarding the user's state, allowing multiple states to occur simultaneously. For instance, a multilabel classification allowed the possibility of a student to be fully engaged while being confused about the learning task.

2. The authors experimented with various baseline approaches on the dataset, namely two types of models; Static Models (frame classification/prediction) and Dynamic Models (temporal video classification). They employed transfer learning and sequence learning separately for video classification. Therefore, their results provided a good baseline for comparing the performance of architectures that combine transfer learning and sequence learning.

The work undertaken by [Schoneveld et al. (2021)], [Abedi and Khan (2021)] and [Schroff et al. (2015)] showed that formulating the detection of learner affective state classification from videos as a spatial-temporal problem would yield better results. Consequently, I began researching an architecture combining transfer and sequence learning. My approach to transfer learning involved developing and training three deep learning models (each based on a paper mentioned above) and building a custom model with an LSTM block followed by Fully connected layers for sequence learning. A block diagram illustrating my work is shown in Figure 3.1. The following sections describe each of my feature extractors in detail.

3.2 Feature Extractor 1: Based on Teacher Network proposed by Schoneveld et al. (2021)

[Schoneveld et al. (2021)] proposed a deep learning approach based on self-knowledge distillation for audio-visual emotion recognition. Knowledge distillation [Hinton et al. (2015)] is a technique in which a large complex model (teacher network) is trained to extract the structure of the dataset and then pass its knowledge to train a smaller network (student network) to match the larger network's prediction on the task of interest. This results in a lighter model with less compute requirements and superior performance, making it easily deployable on edge. The authors used a two-step knowledge distillation process to train their facial expression embedding network (facial feature extractor). The teacher network they implemented was inspired by [Vemulapalli and Agarwala (2019)] and simultaneously trained on two datasets for facial expression embedding extraction:

- **AffectNet dataset** [Mollahosseini et al. (2017)] consists of 450,000 facial images in the wild collected from the Internet and annotated manually for eight facial expression categories (Neutral, Happy, Sad, Surprise, Fear, Disgust, Anger, Contempt) and arousal-valence values. It is the largest database of facial expression, valence, and arousal in the wild.
- **Google Facial Expression Comparison (FEC) dataset** [Vemulapalli and Agarwala (2019)] consists of approximately 700,000 facial image triplets with human annotations that indicate which two faces in each triplet are the most similar.

For AffectNet images, the teacher network's output head predicted the class logits of the input image and for FEC images, the output head of the network found the images that are closest in the learned embedding space. More on this is explained in subsection 3.2.1. Given the time constraint, it wasn't feasible to implement the whole self-distillation process. The paper showed that the teacher network alone matched the self-distillation architecture for AffectNet 8-emotion classification within 0.3% accuracy. As a result, I decided to implement just the teacher network with some modifications.

One might wonder what is the need to train this teacher network on two tasks simultaneously. My rationale to following what the authors had done was due to two reasons: Firstly, [Vielzeuf et al. (2018)] referred in section 2.4 and [Xia and Liu (2015)] referred in section 2.5.1 illustrated the benefit of utilizing additional information in a multi-task learning setup for emotion recognition. It can be thought of as one task complementing the other by adding information. The teacher network was originally presented in the Google FEC paper for performing facial similarity tasks. As I was going to use the model in the emotional recognition space, it made sense to fine-tune it on AffectNet 8-emotion classification tasks to generate a more efficient embedding.

3.2.1 Architecture:

FaceNet [Schroff et al. (2015)] pre-trained on the VGGFace2 [Cao et al. (2018)] dataset is taken up to the Inception 4e block and used as the base model for this feature extractor. The model architecture and weights were obtained from the 'facenet-pytorch' python library [Esler (2020)]. The base model is followed by a 1x1 convolution layer and a series of five untrained DenseNet [Huang et al. (2017)] blocks. Another 1x1 convolution layer with a global average pooling layer is used to reduce the dimension of the output to a 128-dimensional feature embedding, D_{face}. Two independent linear output heads are added for training - a 32-dimensional embedding for the FEC triplets task and an 8-dimensional output head producing class logits for the AffectNet classification task. The network architecture is shown in Figure 3.2. The model is trained to minimise both the AffectNet loss and Google FEC losses simultaneously. The training procedure of the feature extraction network is described in Algorithm 3.1. This network was implemented from scratch with no publicly available code.

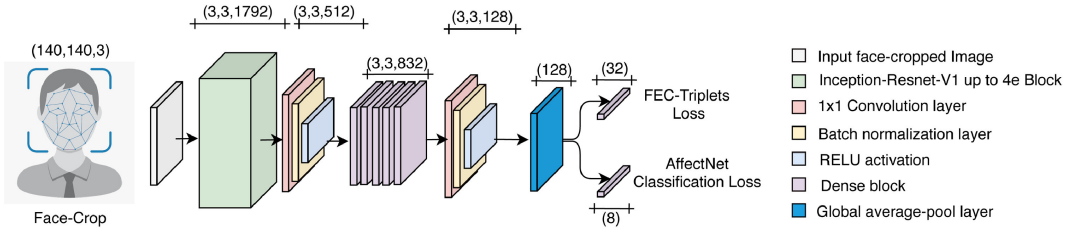


FIGURE 3.2: Feature Extractor 1 based on Teacher Network proposed by et al. The numbers over each block represent the tensor output shape after applying that block.(Source:Schoneveld et al. (2021))

Details of the FaceNet architecture is elaborated in Section 3.4. DenseNet architecture is described in the following section.

3.2.1.1 DenseNet

In Standard ConvNet, input image goes through multiple Convolution layers to obtain high-level features as shown in Figure 3.3. In DenseNet, each layer obtains additional

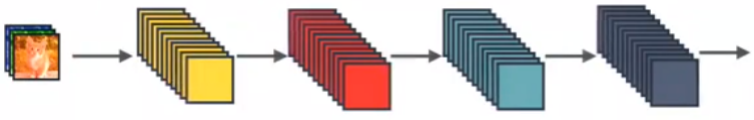


FIGURE 3.3: Standard Convolution network (Source:Tsang (2019))

inputs from all preceding layers and passes on its feature maps to all the subsequent layers as shown in Figure 3.4. Each layer receives a "collective knowledge" from all the preceding layers.

Algorithm 3.1 Given feature network $\mathbf{f}_\Theta$, Google FEC output head $\mathbf{g}_\phi$, AffectNet output head $\mathbf{h}_\theta$, number of training steps $\mathbf{N}$, AffectNet loss weight α

for iterations in the range(N) **do**

$(\mathbf{X}_{FEC}, \mathbf{y}_{FEC}) \leftarrow$ batch of Google FEC triplets and labels

$(\mathbf{X}_{Aff}, \mathbf{y}_{Aff}) \leftarrow$ batch of AffectNet images and class labels

$\mathbf{e}_{FEC} \leftarrow \mathbf{f}_\Theta(\mathbf{X}_{FEC})$ ▷ Face embeddings for FEC images

$\mathbf{e}_{Aff} \leftarrow \mathbf{f}_\Theta(\mathbf{X}_{Aff})$ ▷ Face embeddings for AffectNet images

$\mathbf{v}_{FEC} \leftarrow \mathbf{g}_\phi(\mathbf{e}_{FEC})$ ▷ Predict vectors for triplet loss

$\mathbf{p}_{Aff} \leftarrow \mathbf{h}_\theta(\mathbf{e}_{Aff})$ ▷ Predict class probabilities for AffectNet

$L_{FEC} = triplet\_loss(\mathbf{v}_{FEC}, \mathbf{y}_{FEC})$

$L_{Aff} = cross\_entropy\_loss(\mathbf{p}_{Aff}, \mathbf{y}_{Aff})$

$L = L_{FEC} + \alpha * L_{Aff}$ ▷ Total loss for training step

$\Delta_{all} = \left(\frac{\partial L}{\partial \Theta}, \frac{\partial L}{\partial \phi}, \frac{\partial L}{\partial \theta}\right)$ ▷ Obtain all gradients

$(\Theta, \phi, \theta) \leftarrow Adam(\Delta_{all})$ ▷ Update feature extractor and the output heads' parameters simultaneously

end for

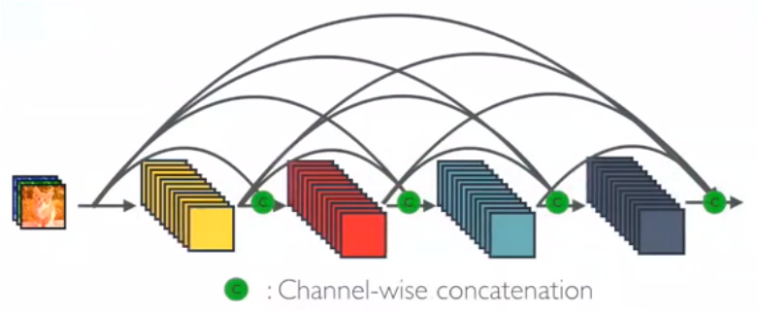


FIGURE 3.4: A Single DenseNet block (Source:Tsang (2019))

Batch normalisation (BN), ReLU and (3×3) Convolution are performed for each composition layer to transform the input, say x_0, x_1, x_2, x_3, to a k-channel x_4 output as shown in Figure 3.5. BN-ReLU-(1×1)Convolution is applied before each BN-ReLU-(3×3)Convolution block to reduce the model complexity and size.

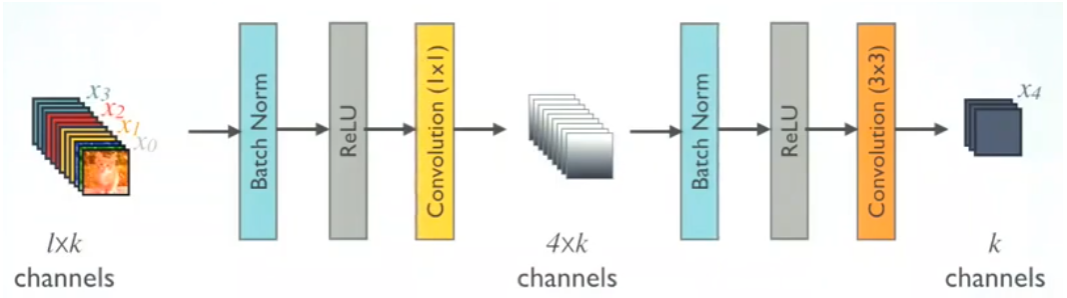


FIGURE 3.5: DenseNet bottleneck layers (Source:Tsang (2019))

3.3 Feature Extractor 2: ResNet18 [He et al. (2016)]

ResNet-TCN model proposed by [Abedi and Khan (2021)] extracted spatial features from video frames using ResNet18 and analysed temporal changes in video frames using Temporal CNN[Bai et al. (2018)]. In their experiment, the authors attempted to detect the engagement state alone and achieved a 39% improvement in accuracy. This inspired me to pair Resnet18 with LSTM.

3.3.1 Resnet18 architecture

ResNet stands for residual networks and was introduced in the paper 'Deep Residual Learning for Image Recognition [He et al. (2016)]'. The residual network has multiple variations, namely ResNet16, ResNet18, ResNet34, ResNet50, ResNet101, ResNet110, ResNet152, ResNet164, ResNet1202, and so forth. ResNet18 is a 72-layer architecture with 18 deep layers. This network's architecture enables large amounts of convolutional layers to function efficiently. However, adding multiple deep layers to a network often degrades the output, known as the vanishing gradient problem. A neural network, during training, performs gradient descent where it iteratively evaluates the parameters, computes the loss and then takes steps in the direction that will minimise the loss. The direction and steps are determined by calculating the gradient and backpropagating it to the prior layers. Due to the presence of multiple layers, the repeated multiplication results in the gradient becoming smaller and smaller, leading to a saturation in the network performance and learning. The primary idea of ResNet is the use of jumping or skip connections which are nothing but identity connections between the layers, as shown in Figure 3.6. Identity connections function as gradient superhighways which

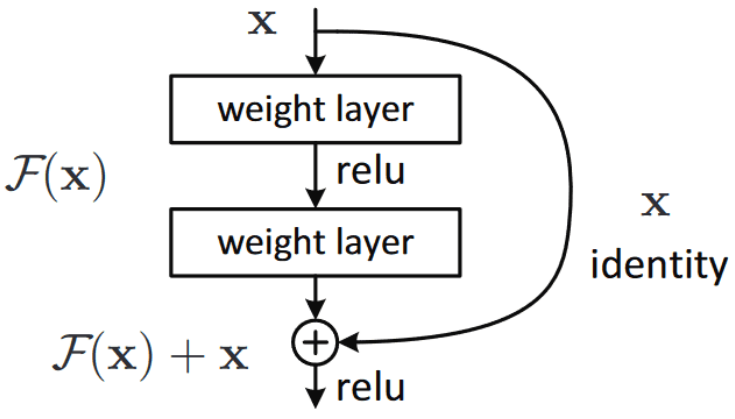


FIGURE 3.6: Skip Connections in a ResNet block (Source: He et al. (2016))

allow the gradient to flow unhindered making it possible for gradients to propagate to deep layers before they are attenuated to small or zero values. The complete ResNet18 architecture is shown in Figure 3.7.

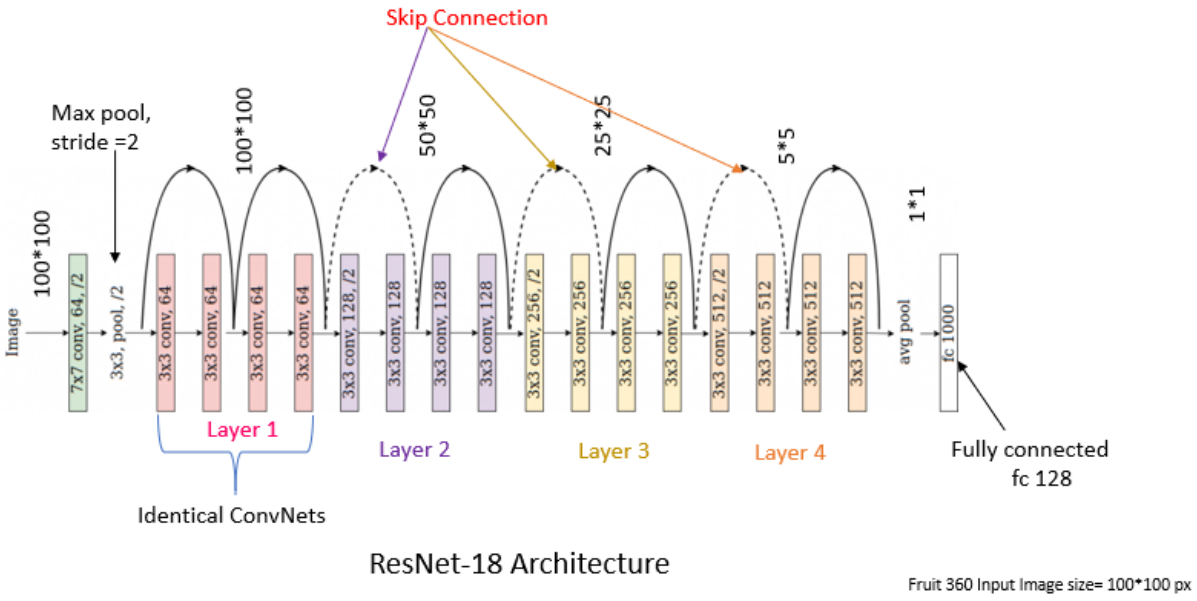


FIGURE 3.7: ResNet18 Architecture (Source:Singhal (2020))

3.3.2 Implementation

I've used ResNet18 implementation available in Pytorch `https://pytorch.org/vision/0.8/models.html#torchvision.models.resnet18` upto the Average Pooling layer. This ResNet model has been pretrained on ImageNet [Deng et al. (2009)]. It returns a feature embedding of dimension (1, 512).

3.4 Feature Extractor 3: Inception-ResNet v1 [Szegedy et al. (2015)] inspired from FaceNet [Schroff et al. (2015)] paper

FaceNet is a state-of-art face recognition, verification and clustering neural network developed by Google that achieved a record accuracy of 99.63% on the Labeled Faces in the Wild (LFW) dataset [Huang et al. (2008)] and 95.12% on YouTube Faces DB [Wolf et al. (2011)].

The building blocks of FaceNet are shown in Figure 3.8. It comprises a 22-layer deep neural network followed by an L2 normalisation layer that converts a 512-dimensional vector to a 128-dimensional feature embedding. The loss function used at the last layer is called triplet loss. It has been one of the most popular loss functions for supervised similarity or metric learning ever since its introduction in the FaceNet paper. To compute

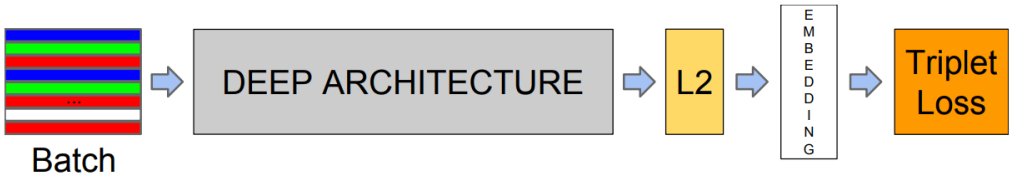


FIGURE 3.8: FaceNet:High level block diagram (Source:[Schroff et al. (2015)])

Triplet loss, a reference input (called anchor) is compared to a matching input (called positive) and a non-matching input (called negative) and the objective of the model is to minimise the distance from the anchor to the positive, and maximise the distance from the anchor to the negative input. FaceNet paper considers the deep network as a black box. It doesn't deal much with the internal workings of the deep neural network, only stating that the deep network block is similar to the GoogleNet architecture. GoogleNet has many versions, but 'Inception-Resenet-v1' [Szegedy et al. (2015)] is the one that I have used.

3.4.1 GoogleNet versions and Inception-Resenet-v1 architecture

GoogleNet won the ImageNet 2014 challenge and is well known for overcoming the following challenegs in conventional CNN:

- A network with more layers is always better, but it also increases the number of parameters and may lead to overfitting.
- Deep networks also suffer from the vanishing gradient problem, where the gradients end up exploding or shrinking to a minimal value, and the model ceases to learn. Sigmoid activation functions often cause this behaviour. During backpropagation, the gradient does not propagate through the network until the initial layer, resulting in unchanged weights.
- A linear increase in filters causes a quadratic increase in operations, requiring more computational power.
- Training more parameters requires large datasets and more time on training; even augmentation of the data won't help much.
- In convolution, as we go deep in the network, the dimension of the input reduces, and information decay happens; therefore, the information extraction should be effective with each passing layer. CNNs require a reduction in representation bottleneck, which calls for a trade-off between dimension reduction and information extraction.

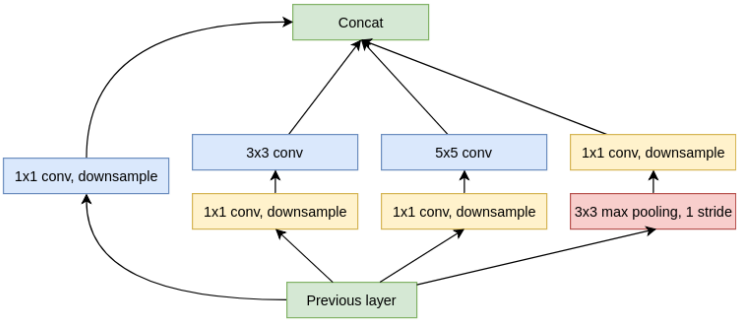


FIGURE 3.9: Inception v1 block with dimensionality reduction(Source: [Szegedy et al. (2015)])

The basic building block of the 'Inception module' is shown in Figure 3.9. GoogleNet uses a 1x1 filter for dimension reduction. With 1x1 convolution, the input size (height and width) remains the same, but the channels are shrunk ie. for example, a 256×256×3 RGB image reduces to 256×256×1 image. In addition, 3×3, 5×5 and 7x7 convolution and 3×3 max-pooling are performed in parallel. All intermediate outputs are concatenated for the next stage, making the inception module broader in the middle but adequately deep since it's connected back to back. The idea behind convolution filters of different sizes is that they will better handle objects at multiple scales. With the depth of the network, back-propagation was bound to cause vanishing gradients; therefore, two auxiliary outputs are tapped at the middle layers and weighted before being added to the total loss as shown in Figure 3.10. This architecture takes image of size 224×224 and all

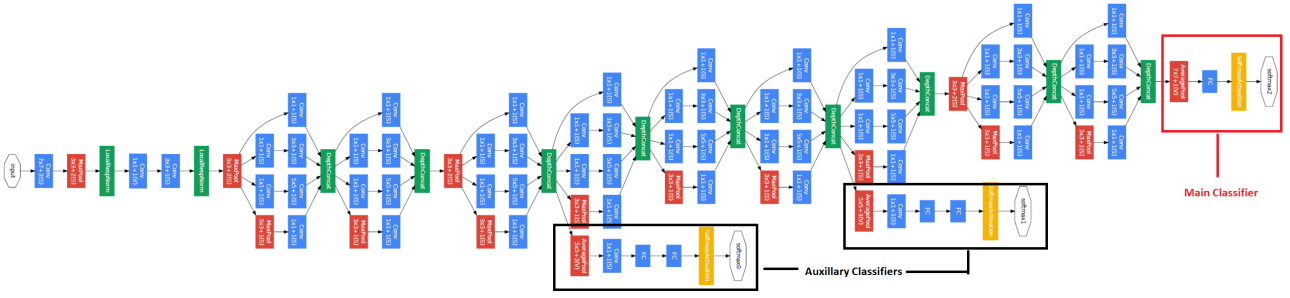


FIGURE 3.10: Full Inception V1 architecture (Source: [Szegedy et al. (2015)])

the convolution layers use Rectified Linear Units (ReLU) as their activation functions.

In the Inception-v2 and Inception-v3 versions, batch normalisation was introduced, factorisation reduced parameters and overfitting, and label smoothing prevented a particular logit from becoming too large compared to others, thus regularising the classifier layer. In Inception-ResNet-v1, the inception blocks are same as before just that they are named as A, B, C. Residual connections were introduced, replacing pooling from the inception module and allowing gradients to flow through a network directly without passing through non-linear activation functions. Non-linear activation functions cause the gradients to explode or vanish.

3.4.2 Implementation

The authors have not published the architecture's code. However, David Sandberg's FaceNet implementation in TensorFlow [sandberg (2017)] using the 'Inception-ResNet-v1' version is the most popular. I have used the PyTorch implementation of this code available in the facenet-pytorch package [Esler (2020)]. The pytorch model weights in the package are initialised using parameters ported from David Sandberg's TensorFlow facenet repository.

3.5 Sequence Learning with Long Short-Term Memory model

Traditionally, neural networks receive a fixed-size input and determine their output based on that input. For example, if we feed a network with a single image, we only receive classification results for that image. With a Recurrent Neural Network (RNN), we can operate on sequences of data vectors where the outputs also depend on previous classification results. However, recurrent networks have short-term memory, so as the gap between the existing data and the previous outputs increases, they lose their effectiveness. Essentially, it is a form of the vanishing / exploding gradient problem, in which the error multiplies with every time step and it becomes more difficult to train the network. Long Short-Term Memory (LSTM) architecture overcomes the problem of missing long-term dependencies in RNNs by making each neuron in the hidden layer perform four operations instead of one.

LSTM has internal mechanisms called gates introduced by various activation layers that regulate the flow of information. During training, these gates determine which data in a sequence should be kept and which should be discarded, allowing the network to pass only relevant information down the long chain of sequences. Figure 3.11 shows a representation of unrolled LSTM cell. Each LSTM recurrent unit has two inputs:

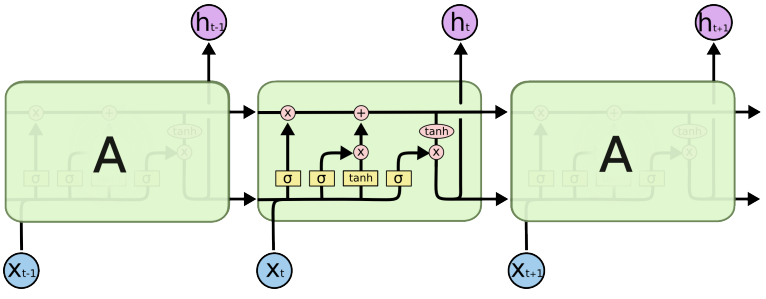


FIGURE 3.11: Representation of unrolled LSTM cell. (Source: https://colah.github.io/)

an input at time-step t, x_t and a hidden state, consisting of the cell state C_{t1}. It also maintains a vector called the Internal Cell State which describes the information retained by the previous LSTM unit. LSTM unit has four gates serving various purposes. The

Forget Gate determines which parts to forget from our long-term memory C_{t1}. It is a dot product of the current cell state and the output of a small, densely connected neural network combining a weighted combination of the last hidden state and the current input as represented by Equation 3.1.

$$f_t = \sigma(W_f \cdot [h_{t1}, x_t] + b_f) \tag{3.1}$$

Input Gate determines what information will be written into Internal Cell State as shown in Equation 3.2.

$$i_t = \sigma(W_i \cdot [h_{t1}, x_t] + b_i) \tag{3.2}$$

By adding non-linearity to the information and making it zero-mean, the input modulation gate modulates information that will be written to the Internal State Cell, represented by Equation 3.3. This reduces the learning time as zero-mean input converges faster. It is a good practice to include this gate in LSTM's structure, even though its actions are not as important as those of the others.

$$\tilde{C}_t = \sigma(tanh(W_c) \cdot [h_{t1}, x_t] + b_c) \tag{3.3}$$

Combining Equations 3.2 and 3.3 creates the new cell state C, by forgetting information using f_t and introducing new memories with $i_t * \tilde{C}_t$

$$\tilde{C}_t = f_t * C_{t1} + i_t] * \tilde{C}_t) \tag{3.4}$$

The Output Gate determines the output (next Hidden State) from the current Internal Cell State through a combination of the our new cell state C_t and the inputs $[h_{t1}, x_t]$.

$$o_t = \sigma(W_o \cdot [h_{t1}, x_t] + b_o) \tag{3.5}$$

$$h_t = o_t * tanh(C_t) \tag{3.6}$$

For sequence learning I implemented a custom two-layer LSTM block in Tensorflow. The following section explains the implementation of the LSTM blocks as well as the entire training pipeline.

3.6 End-to-end flow and training pipeline

Frames were extracted from the DAiSEE video snippets at the rate of 30 fps using the ffmpeg pacakge [Tomar (2006)]. From these frames faces were detected, cropped and aligned using Multi-Task Cascaded Convolutional Neural Networks (MTCNN)[Zhang et al. (2016)] available in the facenet-pytorch package as existing literature showed it

Layer (type)	Output Shape	Param	Connected to
lstm_input(InputLayer)	[(None, 290, 128)]	0	[]
lstm (LSTM)	(None, 290, 256)	394240	[lstm_input[0][0]]
dropout (Dropout)	(None, 290, 256)	0	[lstm[0][0]]
lstm_1 (LSTM)	(None, 256)	525312	[dropout[0][0]]
fc1 (Dense)	(None, 128)	32896	[lstm_1[0][0]]
fc2 (Dense)	(None, 64)	8256	[fc1[0][0]]
y1 (Dense)	(None, 4)	260	[c2[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]]

TABLE 3.1: Sequence Learning Model Summary

had the fastest detection rate. The cropped faces were then fed to one of the three feature extractors to extract sequences of feature embeddings. This resulted in feature sequences of dimensions (300, D_{face}) for each video, where D_{face} indicates the output embedding dimension from each feature extractor. The features from each extractor was stored in numpy files for sequence learning. The features from the numpy files were fed to the LSTM block. I implemented a two-layer LSTM block with hidden and output dimensionality of 256 with a dropout layer in between with a dropout rate of 0.3. The input to the block is an B $\times$ S $\times$ D tensor, where B, S, and D correspond to batch size, the number of frames(sequence length) and embedding dimension respectively. This is followed by two full connected layers with 512 and 128 nodes and relu activation function. There are four output heads placed after the full connected layers to predict the level of each of the four affective states. The model summary is given in table 3.1

Chapter 4

Experiments, Results and Discussion

Implementation details and experiments conducted as part of the project have been discussed in this chapter. Furthermore, it comprehensively analyses the results and shows how my approach compares to existing methods.

4.1 DAiSEE Dataset Analysis

4.1.1 Data Imbalance

The end-to-end FeatureExtractor+LSTM pipeline is trained and evaluated on the DAiSEE video dataset. The dataset contains 9068 in-the-wild videos captured from 112 students watching an online course recording, to recognise the affective states of Boredom, Confusion, Engagement, and Frustration. Each state is annotated for four intensity levels as 0(very low), 1(low), 2(high), and 3(very high). The dataset comprises 10-second videos of 30 fps and 640 × 480 resolution.

Table 4.1 shows the distribution of samples in the train, validation, and test sets and Figure 4.1 illustrates this distribution of labels. It can be easily understood that the

	Train Set				**Validation Set**				**Test Set**			
State	**B**	**E**	**C**	**F**	**B**	**E**	**C**	**F**	**B**	**E**	**C**	**F**
Level 0	2433	34	3616	4183	446	23	942	1058	823	4	1200	1388
Level 1	1696	213	1245	941	376	143	322	271	584	84	427	316
Level 2	1073	2617	431	191	475	813	153	81	338	882	136	57
Level 3	156	2494	66	43	132	450	12	19	39	814	21	2
Total	5358				1429				1784			

TABLE 4.1: Distribution of samples in Train, Validation and Test set.

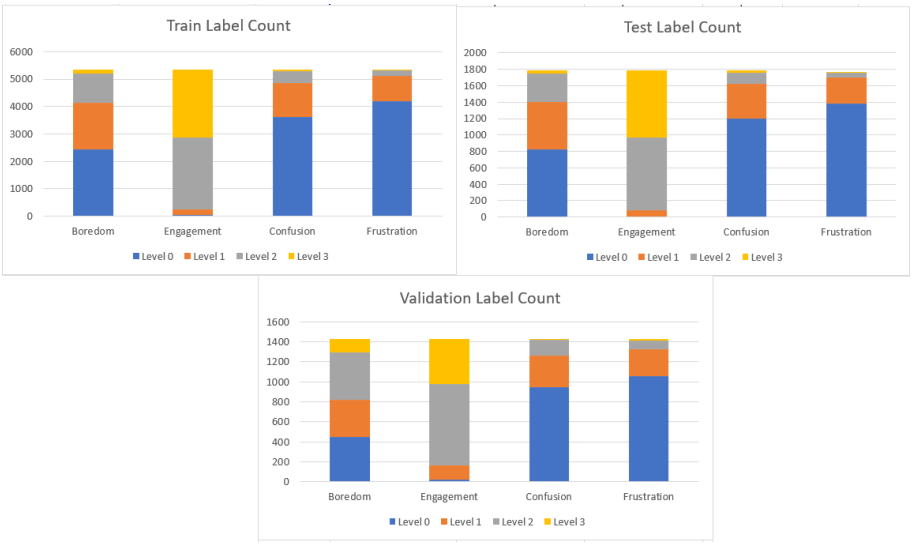


FIGURE 4.1: Demonstration of Data Imbalance

dataset is highly imbalanced. For example, in the case of the engagement state, only 0.63% of the train set, 1.61% of the validation set, and 0.22% of the test set belong to level 0 (highlighted in yellow), while 49% of the train set, 56% of the validation set and 50% of test set belong to level 2 (highlighted in red). Later in Section 4.3 we will see confusion matrices that show how this data imbalance has drastically affected the precision of my models by injecting bias towards the majority classes. The data imbalance can be handled through weighted sampling and weighted loss, which I couldn't implement due to time constraints. But more on this is discussed in Chapter 5.

4.1.2 Frame Similarity

In my initial evaluations, I extracted frames at a rate of 30 fps, resulting in 300 frames per video. Each sequence of 512-dimensional feature embedding had a dimension of (300,512) and was a 600KB NumPy file, so feature extraction consumed a lot of storage space and time. The fact that '*A reduction in the number of frames might not cause a drastic loss of information if the frames are similar*' did not occur to me until one of my supervisors pointed it out. Therefore, to determine the similarity between frames, I sampled 10 frames and checked their cosine similarity. Cosine similarity measures the cosine of the angle between two vectors representing the images in the feature space. I computed it using Scipy's spatial.distance.cosine() and equation 4.1:

$$CosineSimilarity = -1 * (CosineDistance(Frame1, FrameN) - 1) \tag{4.1}$$

Figure 4.2 displays the 10 samples and their cosine similarity score to the first frame. It turned out to be quite similar; for instance, frame 1 and frame 60 had a similarity of 0.98, indicating they are 98% similar. This prompted me to evaluate my model's performance

with 100 frames per video. 100 frames reduced the storage size of one video sequence to 200KB. Also, reducing the number of frames eliminated redundant information. More

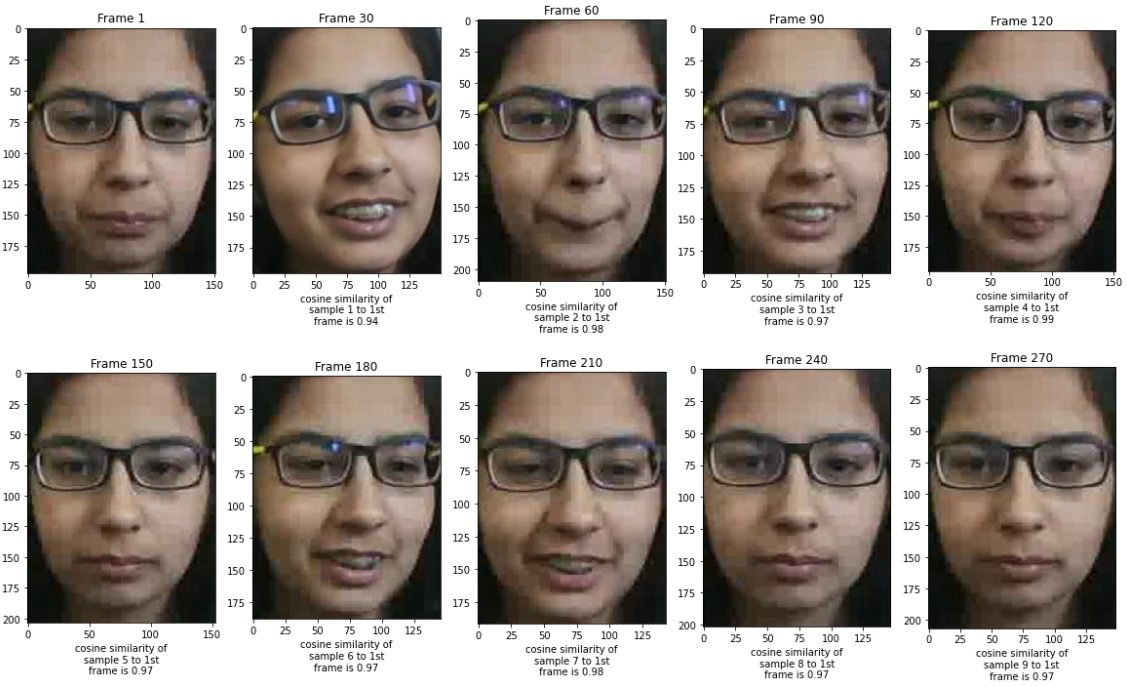


FIGURE 4.2: Cosine similarity between 10 samples of a video clip

about the performance is given in Section 4.3.

4.2 Experiment Settings

The videos were down-sampled, temporally and spatially, to get (B×C×H×W) tensors as inputs to the feature extractor architectures, where B denotes the batch size, C stands for the nunmber of channels and (H×W) represents the size of the frame. For training the teacher network based feature extractor, AffectNet images and Google FEC image triplets were resized to (3,140,140), which is the standard input dimension of the FaceNet base model. Depending on the task the image belonged to, this network extracted 128-dimensional feature embeddings from the input images and fed them to the respective output heads. Adam was the optimiser used with a learning rate of 0.005 and batch size of 64. Early stopping was employed for a Patience of 10. During training, the AffectNet cross-entropy loss is multiplied by $\alpha = 0.1$ to match the Google FEC triplet loss' magnitude as done by the authors. In their paper, the authors stated that training the model for 13 epochs produced the best validation performance; hence the epochs were set to 13.

ResNet-18 and Inception-ResNet-V1 extractors accepted an input dimension of (3×224×224) and (3×160×160) respectively. Both of them extract feature vectors of dimension 512

Exp No.	Model	Train Data size	No. of frames	D_face size	batch size	epochs	LSTM Block	FC Layers	Comments
Exp 1	Feature Ext 1 (Teacher Network) + LSTM	1430	300	128	32	100	Layer 1: 256 Layer 2: 256 Dropout: 30%	FC1:128 FC2: 64	
Exp 2	Feature Ext 1 (Teacher Network) + LSTM	1430	300	128	32	100	Layer 1: 256 Layer 2: 256 Dropout: 30%	FC1: 512 FC2:128	FC Layers modified
Exp 3	Feature Ext 2 (ResNet-18) + LSTM	1430	300	512	32	100	Layer 1: 256 Layer 2: 256 Dropout: 30%	FC1: 512 FC2:128	Embedding size Changes
Exp 4	Feature Ext 3 (Inception-ResNet) + LSTM	1430	300	512	32	100	Layer 1: 256 Layer 2: 256 Dropout: 30%	FC1: 512 FC2:128	
Exp 5	Feature Ext 3 (Inception-ResNet) + LSTM	1420	100	512	32	100	Layer 1: 256 Layer 2: 256 Dropout: 30%	FC1: 512 FC2:128	Frames count modified
Exp 6	Feature Ext 3 (Inception-ResNet) + LSTM	5343	100	512	32	100	Layer 1: 256 Layer 2: 256 Dropout: 30%	FC1: 512 FC2:128	Train data size increased
Exp 7	Feature Ext 3 (Inception-ResNet) + LSTM	1420	100	512	32	100	Layer 1: 512 Layer 2: 1024 Dropout: 50%	FC1: 512 FC2:128	LSTM modified
Exp 8	Feature Ext 3 (Inception-ResNet) + LSTM	5343	100	512	32	100	Layer 1: 512 Layer 2: 1024 Dropout: 50%	FC1:512 FC2:128	Increased Train data and modified LSTM
Exp 9	Feature Ext 3 (Inception-ResNet) + LSTM	5343	100	512	32	100	Layer 1: 512 Layer 2: 1024 Dropout: 50%	FC1:1024 FC2:512	Increased Train data, fine tuned LSTM, fine tuned FC layers
Exp 10	Feature Ext 3 (Inception-ResNet) + LSTM	5343	100	512	16	100	Layer 1: 512 Layer 2: 1024 Dropout: 50%	FC1: 1024 FC2:512	Batch size reduced
Exp 11	Feature Ext 3 (Inception-ResNet) + LSTM	5343	100	512	16	100	Layer 1: 512 Layer 2: 1024 Dropout: 50%	FC1:512 FC2:128	Batch size reduced and FC modified

TABLE 4.2: Experiments conducted to evaluate the model performance

from consecutive frames. All three feature extractors are implemented using the PyTorch framework.

Sequences of feature embeddings from each feature extractor were saved to NumPy files and later fed into the sequence learning architecture as (B×S×D_{face}) tensors where B denotes the batch size, S denoted the frame count and D_{face} represents the face embedding dimension. Parameter optimisation for sequence learning is achieved using Adam with a learning rate of 0.001. The sequence learning block is implemented using Tensorflow due to the ease of implementation. All the associated codes of this project are available at `https://github.com/Resh-97/StudentEngagementDetection`.

Table 4.2 summarises all the experiments that were conducted to evaluate the model performances. These experiments were conducted to comprehend how the model performance is influenced by the train data size, frame count, fine-tuning fully connected layers, fine-tuning LSTM blocks and the batch size.

4.2.1 Training and Evaluation Metrics

During training, the sequence model attempts to reduce the **Sparse Categorical Cross-Entropy** loss function for each recognition task. Cross-entropy is a measure of the difference between probability distributions of different categories. In categorical cross-entropy, true labels are one-hot encoded, while in sparse categorical cross-entropy, the true labels are encoded as integers like [0], [1], [2] and [3] for a four-class problem like mine. Sparse categorical cross entropy saves time and memory because it uses a single integer instead of a whole vector for a class.

Most papers and models that employed the DAiSEE dataset for evaluation have only reported the model's accuracy, with little to no insights into the precision or F1 scores. However, for an imbalanced dataset this measure can often be misleading. It is challenging for a model to learn the features of an infrequently encountered class. Therefore, the model will skew predictions towards more common categories to improve the metrics used for training. When applying the model to unseen data with varying label distribution, this becomes an issue. Additionally, if the disparity in the number of samples in each category is vast, the model might fetch a very high accuracy even when one or more classes are never detected. Therefore, I have also computed the 'F1-Score' and 'Confusion Matrix' for each evaluation run to provide further feedback on model behaviour.

4.2.2 Computing Services and File-stores

Given the size of the models and amount of data, the models were run on the the University High Performance Computing service - Iridis 5. Iridis 5 uses the SLURM resource manager to run jobs on the main compute clusters. For the project, I used the Lyceum partition that offers 4 Consumer GTX1080Ti GPU nodes with a walltime of 60 hours. The datasets, scripts, model weight checkpoints and results were stored primarily on the two available file systems on Iridis: /mainfs and /scratch. /mainfs and /scratch have a hard data limit of 130 GB and 1500 GB respectively. However, towards the last few days of my research I was also forced to use the shared solid-state /ssdfs partition of size 57 TB for storage.

The GPU cluster takes a while to return the outputs and the logs cannot be viewed in real-time during execution, which makes bug-fixing a slow process. Therefore, I also used 'Google Colab' in conjunction for debugging. All the frame extraction scripts were executed in my local system. This was done to save time as every dataset uploaded onto the HPC filestore had to be compressed and extracted costing time and storage space.

	Boredom				
	Accuracy	F1 Score			
Exp No.		Level 0	Level 1	Level 2	Level 3
Exp 1	19%	NA	NA	NA	NA
Exp 2	41.33%	0.60	0	0.17	0
Exp 3	46.34%	0.63	0	0	0
Exp 4	45.16%	0.59	0.25	0	0
Exp 5	33.37%	0.12	0.49	0	0
Exp 6	42.36%	0.51	0.42	0.27	0
Exp 7	42.82%	0.59	0.18	0.21	0
Exp 8	39.08%	0.43	0.40	0.35	0
Exp 9	46.55%	0.64	0	0	0
Exp 10	32.24%	0.42	0.10	0.32	0
Exp 11	35.12%	0.47	0.09	0.34	0

TABLE 4.3: Accuracy and F1 score for Boredom state

	Engagement				
	Accuracy	F1 Score			
Exp No.		Level 0	Level 1	Level 2	Level 3
Exp 1	49.60%	NA	NA	NA	NA
Exp 2	49.61%	0	0	0.66	0
Exp 3	49.61%	0	0	0.66	0
Exp 4	54.22%	0	0	0.67	0.29
Exp 5	52.88%	0	0	0.67	0.22
Exp 6	57.58%	0	0	0.67	0.43
Exp 7	48.70%	0	0	0.56	0.41
Exp 8	57.30%	0	0	0.68	0.39
Exp 9	49.72%	0	0	0.66	0
Exp 10	56.33%	0	0	0.65	0.46
Exp 11	56.90%	0	0	0.66	0.45

TABLE 4.4: Accuracy and F1 score for Engagement state

4.3 Results and Discussions

This section analyses the results of the 11 Experiments mentioned in Section 4.2, conducted to evaluate the performance of the models. The results are consolidated in Table 4.3, Table 4.4, Table 4.5 and Table 4.6 for ease of reference. Each affective state is benchmarked individually for each experiment.

4.3.1 Comparison of the Vanilla architecture of the proposed methods to Baselines models on DAiSEE

The authors of the DAiSEE dataset [Gupta et al. (2016)] have implemented two video-level classification methods: C3D transfer learning [Tran et al. (2015)] and Long-Term Recurrent Convolutional Network (LRCN) [Gupta et al. (2016)], of which they stated

	Confusion				
	Accuracy	**F1 Score**			
Exp No.		**Level 0**	**Level 1**	**Level 2**	**Level 3**
Exp 1	67.50%	NA	NA	NA	NA
Exp 2	67.45%	0.81	0	0	0
Exp 3	67.45%	0.81	0	0	0
Exp 4	67.45%	0.81	0	0	0
Exp 5	67.42%	0.81	0	0	0
Exp 6	67.48%	0.81	0.01	0	0
Exp 7	66.52%	0.79	0.15	0	0
Exp 8	67.42%	0.81	0.01	0	0
Exp 9	67.42%	0.81	0	0	0
Exp 10	67.42%	0.81	0	0	0
Exp 11	67.42%	0.81	0.02	0.02	0

TABLE 4.5: Accuracy and F1 score for Confusion state

	Frustration				
	Accuracy	**F1 Score**			
Exp No.		**Level 0**	**Level 1**	**Level 2**	**Level 3**
Exp 1	77.93%	NA	NA	NA	NA
Exp 2	77.93%	0.88	0	0	0
Exp 3	77.93%	0.88	0	0	0
Exp 4	77.93%	0.88	0	0	0
Exp 5	77.94%	0.88	0	0	0
Exp 6	77.94%	0.88	0	0	0
Exp 7	77.94%	0.88	0	0	0
Exp 8	77.94%	0.88	0	0	0
Exp 9	77.94%	0.88	0	0	0
Exp 10	77.94%	0.88	0	0	0
Exp 11	77.94%	0.88	0	0	0

TABLE 4.6: Accuracy and F1 score for Frustration state

Affective State	C3D Transfer learning after fine-tuning	LRCN	Feature Extractor 1 (Teacher Network) + LSTM [Experiment 2]	Feature Extractor 2 (ResNet18) + LSTM [Experiment 3]	Feature Extractor 3 (Inception-ResNetV1) + LSTM [Experiment 4]
Boredom	**45.2**	**53.7**	**41.3**	**46.3**	**45.2**
Engagement	**56.1**	**57.9**	**49.6**	**49.6**	**54.2**
Confusion	**66.3**	**72.3**	**67.5**	**67.5**	**67.5**
Frustration	**79.1**	**73.5**	**77.9**	**77.9**	**77.9**

TABLE 4.7: Performance Comparison of Vanila architecture of the proposed methods to Baseline models

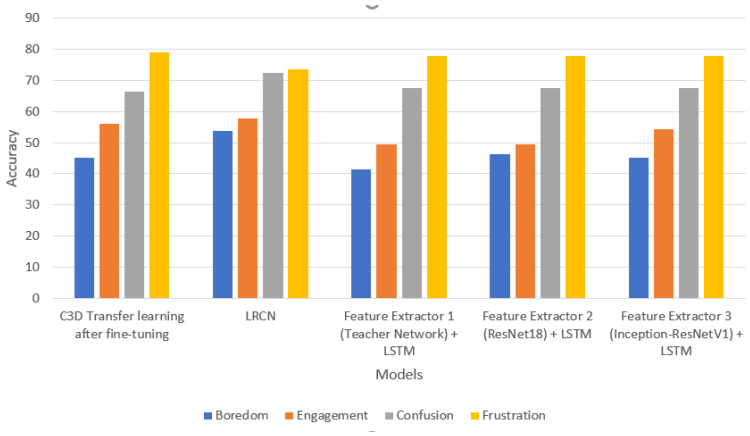


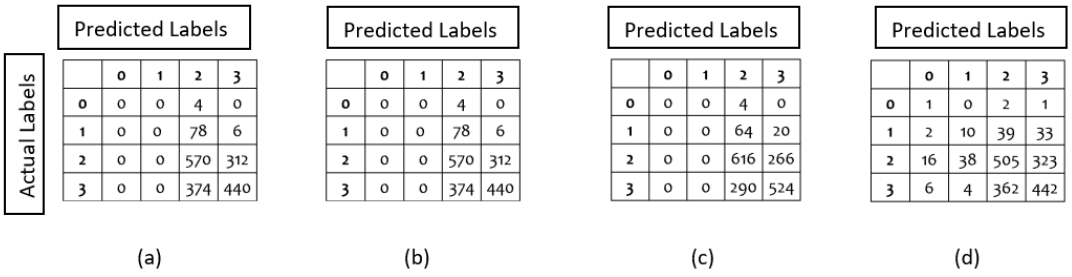
FIGURE 4.3: Performance Comparison of Vanila architecture of the proposed methods to Baseline models. Refer. Table 4.7 for accuracy values.

LRCN generally performed better than all the other methods. Figure 4.3 and Table 4.7 highlight how the vanilla architectures of the proposed models stand against the above baselines. The vanilla architectures of [Feature Extractor 1 + LSTM], [Feature Extractor 2 + LSTM] and [Feature Extractor 3 + LSTM] have the specifications of Experiment 2, Experiment 3 and Experiment 4, respectively. The baseline models were trained on 5358 train videos and 1429 validation videos while tested on 1784 test videos. However, despite being trained on only 1429 videos due to storage constraints, my vanilla architectures often slightly outperform or deliver comparable results to the baselines.

Generally, the models employing Feature Extractors 2 and 3 perform better than the C3D transfer learning model. In contrast, they either provide similar results to the LRCN model or struggle to outperform it (while within 8%, 4% and 5% accuracy for boredom, engagement and confusion states, respectively). It may be due to the fact that the LSTM blocks are not customised to each feature extractor in the vanilla versions. This, however, doesn't seem to affect the frustration state, for which the LRCN method performs the worst. That is because the expressions of frustration were often short spanned in the video, ranging only within a few frames and not displayed long which meant that only a few frames encapsulated the frustration expression with not much change temporally. Of the three mental states, boredom state gets the lowest accuracy and this can be attributed to the disparity in distribution of levels within this state.

None of the previous work on the DAiSEE dataset, working on the original four-class annotations, reported their confusion matrices and as explained in subsection 4.2.1 accuracy alone as a performance metric is misleading when the data is imbalanced. Therefore, it was hard to determine the performance of their methods on individual categories. However, two other state-of-the-art papers [Abedi and Khan (2021)] and [Liao et al. (2021)] performing engagement level recognition on the DAiSEE dataset have shed light on the confusion matrix for engagement state. [Abedi and Khan (2021)] proposed ResNet+TCN and achieved an accuracy of 63.1%, while [Liao et al. (2021)] propsed

an LSTM Network with Global Attention (GALN) achieving an accuracy of 58.84% for engagement level recognition; both outperforming the baselines. [Abedi and Khan (2021)] also re-implemented the baselines and made the confusion matrix available in their paper. Figure 4.4 shows the engagement-level confusion matrices of C3D method, LRCN, ResNet+TCN and ResNet+TCN with weighted sampling and weighted loss. Figure 4.5 shows the engagement-level confusion matrices of the experiments conducted in this project.



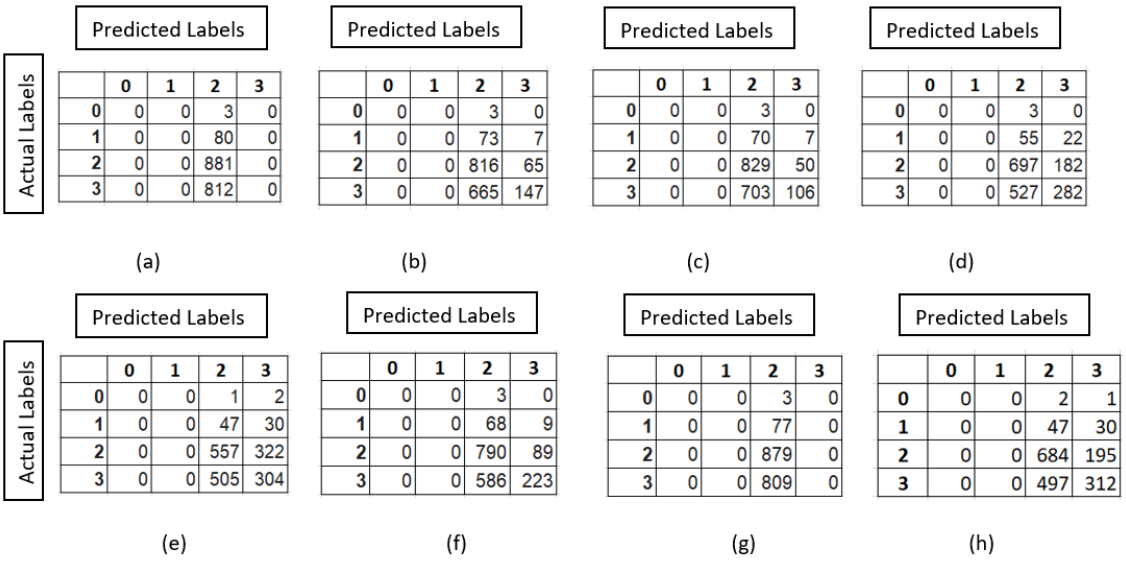
(a)	0	1	2	3
0	0	0	4	0
1	0	0	78	6
2	0	0	570	312
3	0	0	374	440

(b)	0	1	2	3
0	0	0	4	0
1	0	0	78	6
2	0	0	570	312
3	0	0	374	440

(c)	0	1	2	3
0	0	0	4	0
1	0	0	64	20
2	0	0	616	266
3	0	0	290	524

(d)	0	1	2	3
0	1	0	2	1
1	2	10	39	33
2	16	38	505	323
3	6	4	362	442

FIGURE 4.4: Engagement-level confusion matrices of (a)C3D method, (b)LRCN, (c) ResNet+TCN and (d)ResNet+TCN with weighted sampling and weighted loss.



(a)	0	1	2	3
0	0	0	3	0
1	0	0	80	0
2	0	0	881	0
3	0	0	812	0

(b)	0	1	2	3
0	0	0	3	0
1	0	0	73	7
2	0	0	816	65
3	0	0	665	147

(c)	0	1	2	3
0	0	0	3	0
1	0	0	70	7
2	0	0	829	50
3	0	0	703	106

(d)	0	1	2	3
0	0	0	3	0
1	0	0	55	22
2	0	0	697	182
3	0	0	527	282

(e)	0	1	2	3
0	0	0	1	2
1	0	0	47	30
2	0	0	557	322
3	0	0	505	304

(f)	0	1	2	3
0	0	0	3	0
1	0	0	68	9
2	0	0	790	89
3	0	0	586	223

(g)	0	1	2	3
0	0	0	3	0
1	0	0	77	0
2	0	0	879	0
3	0	0	809	0

(h)	0	1	2	3
0	0	0	2	1
1	0	0	47	30
2	0	0	684	195
3	0	0	497	312

FIGURE 4.5: Engagement-level confusion matrices of (a)Experiment 3, (b)Experiment 4, (c) Experiment 5, (d)Experiment 6, (e)Experiment 7, (f)Experiment 8, (g) Experiment 9 and (h)Experiment 11

Due to the highly-imbalanced distribution of samples, none of the methods are able to classify any samples in the first two levels of engagement correctly. [Liao et al. (2021)] reported their results for one fold of cross-validation during training on the DAiSEE dataset. Even though the authors did not report the confusion matrix, they did state that they observed all the samples being classified to the two higher engagement levels

	Accuracy	F1 Score				Accuracy	F1 Score			
Exp No.		**Level 0**	**Level 1**	**Level 2**	**Level 3**		**Level 0**	**Level 1**	**Level 2**	**Level 3**
Exp 8	39.08%	0.43	0.40	0.35	0	57.30%	0	0	0.68	0.39
Exp 9	46.55%	0.64	0	0	0	49.72%	0	0	0.66	0

TABLE 4.8: Effects of modifying FC layers: Comparison of results from experiments 8 and 9

and that their method could not classify samples belonging to the lower engagement levels.

In terms of F1-score, Inception-Resnet-V1 based architecture with specification from Experiment 6 performs the best for all labels. However, in terms of accuracy, the choices vary.

Although the proposed vanilla architectures aren't significantly outperforming the state-of-the-art results, I wanted to convince the readers that it has the potential given certain modifications are made. This was the motive behind Experiments 5-11 on Inception-Resnet-V1 based model. The following subsections highlight how certain modifications in model specifications and parameters can bring performance improvement in my approaches.

4.3.2 Effect of modifying Fully Connected layers

To study the effect of fine-tuning the fully connected layers, I compared Experiment 1 and 2 for Teacher network based model, and Experiment 8 and 9 for Inception-ResNet-V1 based model. In experiment 1, I used FC1 with 128 nodes and FC2 with 64 nodes and in experiment 2, I used FC1 with 512 nodes and FC2 with 128 nodes. There was an accuracy improvement of 22.33% for Boredom and 0.1% for Engagement.

In experiment 8, I used FC1 with 512 nodes and FC2 with 128 nodes and in experiment 9, I used FC1 with 1024 nodes and FC2 with 512 nodes. There was an accuracy improvement of 7.47% for Boredom and decrement of 7.54% for Engagement. Furthermore, the precision for these states dropped and the models always predicted a single level as shown in Table 4.8.

These experiments thus provide a clear evidence of the effect of fine-tuning/customising the fully-connected layers for each feature extractor.

4.3.3 Effects of modifying LSTM block

To study the effect of tweaking and customising the LSTM block, I compared the results from Experiment 6 and 8 conducted on the Inception-ResNet-V1 based model. In Experiment 6, I used two LSTM layers with nodes 256 each and a dropout rate of 30% while in Experiment 8, an LSTM block with the first layer having 512 nodes and second

	Boredom					Engagement				
	Accuracy	F1 Score				Accuracy	F1 Score			
Exp No.		**Level 0**	**Level 1**	**Level 2**	**Level 3**		**Level 0**	**Level 1**	**Level 2**	**Level 3**
Exp 6	42.36%	0.51	0.42	0.27	0	57.58%	0	0	0.67	0.43
Exp 8	39.08%	0.43	0.40	0.35	0	57.30%	0	0	0.68	0.39

TABLE 4.9: Effects of modifying LSTM block: Comparison of results from experiments 6 and 8

	Boredom					Engagement					Confusion				
	Accuracy	F1 Score				Accuracy	F1 Score				Accuracy	F1 Score			
Exp No.		**Level 0**	**Level 1**	**Level 2**	**Level 3**		**Level 0**	**Level 1**	**Level 2**	**Level 3**		**Level 0**	**Level 1**	**Level 2**	**Level 3**
Exp 5	33.37%	0.12	0.49	0	0	52.88%	0	0	0.67	0.22	67.42%	0.81	0	0	0
Exp 6	42.36%	0.51	0.42	0.27	0	57.58%	0	0	0.67	0.43	67.48%	0.81	0.01	0	0
Exp 7	42.82%	0.59	0.18	0.21	0	48.70%	0	0	0.56	0.41	66.52%	0.79	0.15	0	0
Exp 8	39.08%	0.43	0.40	0.35	0	57.30%	0	0	0.68	0.39	67.42%	0.81	0.01	0	0

TABLE 4.10: Effects of increasing train dataset size: Comparison of results from experiments 5,6,7 and 8

layer having 1024 nodes with a dropout of 50%. There was an accuracy improvement of 3.28% for Boredom and 0.28% for Engagement along with an improvement in F1-score.

4.3.4 Effects of increasing the Train Dataset Size

As stated earlier, all my vanilla architectures were trained on only 1429 videos as opposed to the 5343 videos used by existing methods. I wanted to highlight the fact that given more train data the performance of my models will further improve. Therefore, to study the effect of training data size, I compared the results from Experiment 5 and 6, and Experiment 7 and 8 conducted on the Inception-ResNet-V1 based model.

In Experiment 5 and 7, I used 1420 train dataset videos while in Experiment 6 and 8 I used 5343 videos. For Experiment 6, there was an accuracy improvement of 8.99% for Boredom, 4.7% for Engagement and 0.06% for Confusion along with an improvement in F1-score. Additionally, level 2 of Boredom and level 1 of Confusion which was not previously detected were identified. This is because, with more train data, the model began encountering more samples of level 0 and level 1 of these mental states. A similar trend can be observed with Experiments 7 and 8.

4.3.5 Effects of reducing Frame Count

From the cosine similarity scores in Section 4.1.2, it is clear that the frames are highly correlated. My hypothesis was that this injects redundant information that might cause class bias in the model. Additionally, several clips are unrelated to the actual classification (e.g. face is impassive, equivalent to neutral, for most of the videos, with only a few frames showing the labelled emotion). To study the effect of reducing the frame count, I compared the results of Experiments 4 and 5 as shown in Table 4.11

I was surprised to see my hypothesis was proven false (ie.,reduction in frame count the accuracy and F1-score of the states deteriorated). But in these experiments the model

	Boredom					Engagement				
	Accuracy	F1 Score				Accuracy	F1 Score			
Exp No.		Level 0	Level 1	Level 2	Level 3		Level 0	Level 1	Level 2	Level 3
Exp 4	45.16%	0.59	0.25	0	0	54.22%	0	0	0.67	0.29
Exp 5	33.37%	0.12	0.49	0	0	52.88%	0	0	0.67	0.22

TABLE 4.11: Effects of reducing frame count: Comparison of results from experiments 4 and 5

	Boredom					Engagement					Confusion				
	Accuracy	F1 Score				Accuracy	F1 Score				Accuracy	F1 Score			
Exp No.		Level 0	Level 1	Level 2	Level 3		Level 0	Level 1	Level 2	Level 3		Level 0	Level 1	Level 2	Level 3
Exp 9	46.55%	0.64	0	0	0	49.72%	0	0	0.66	0	67.42%	0.81	0	0	0
Exp 10	32.24%	0.42	0.10	0.32	0	56.33%	0	0	0.65	0.46	67.42%	0.81	0.00	0	0

TABLE 4.12: Effects of modifying batch size: Comparison of results from experiments 9 and 10

was trained on only 1429 videos and tested on 1784 videos, so I'm guessing the label infrequency might have caused this to happen. To further validate this hypothesis, one must evaluate the model trained on 5343 videos, which I couldn't perform due to time constraints.

4.3.6 Effects of modifying the batch size

Batch size is one of the most important hyperparameters of deep learning models and plays a huge role in overfitting if not chosen cautiously. To validate this hypothesis and the impact of modifying the batch size I compared the results of Experiments 9 and 10 as given in Table 4.12. For Experiment 9, I used a batch size 32 and for Experiment 10 used a batch size of 16.

It was seen that though the accuracy reduced with reduction in batch size, the F1-scores improved. This meant that the generalisation of my model improved with reduction in batch size.

4.3.7 Best Models for each Learner State

Table 4.13 highlights the experiments that fetched the best results for each learner state. In the case of Boredom, although experiment 9 yielded the highest accuracy of 46.55%, the model could only detect Level 0, indicating high bias. In contrast, experiment 6 reduced the accuracy by 4.19% and demonstrated better generalisability. This is because experiment 9 contains more nodes in the LSTM and FC block, tuning it heavily to the majority class. For Engagement, experiment 6 delivers the best accuracy and F1 score results. For the Confusion class, a reduction in batch size from 32 to 16 improved the precision of the model and allowed more levels to be detected. For Frustration, there wasn't a significant improvement across experiments due to the vast disparity in the distribution of samples.

Mental State	Metric	Best Model	Result					Feature of the model
			Accuracy	Level 0	Level 1	Level 2	Level 3	
Boredom	**F1-score**	Exp 6	42.36%	0.51	0.42	0.27	0	Increase in train data with reduction in frame count
	Accuracy	Exp 9	46.55%	0.64	0	0	0	Same as Exp 6 but more nodes in LSTM and FC layers
Engagement	**Accuracy & F1-score**	Exp 6	57.58%	0	0	0.67	0.43	Increase in train data with reduction in frame count
Confusion	**Accuracy**	Exp 6	67.48%	0.81	0.01	0	0	Increase in train data with reduction in frame count
	F1-score	Exp 11	67.42%	0.81	0.02	0.02	0	Same as Exp 6 with reduction in batch size and more nodes in LSTM
Frustration	**Accuracy & F1-score**	Exp 5-11	77.94%	0.88	0	0	0	Reduction in frame count from 300 to 100

TABLE 4.13: Best Models for each Mental state

4.3.8 Concluding remarks on the model Performances

The models proposed in the paper currently produce results close to the baseline and occasionally even outperforms it. Through the numerous experiments conducted I have successfully shown that the proposed models have the potential to significantly outperform state-of-the art results on the DAiSEE dataset with modifications on train data size, batch size, frame count, LSTM blocks, Fully conected layers and fine-tuning of pre-trained models. Of the vanilla versions of three models, the one based on Inception-ResNet-V1 performed the best and showed better generalisation capability. This is corroborated by the Loss Vs Epochs graph shown in Figure 4.6. However, I refrain from strongly commenting on the performance of these models against each other, as the first two models were not tested extensively as the third one.

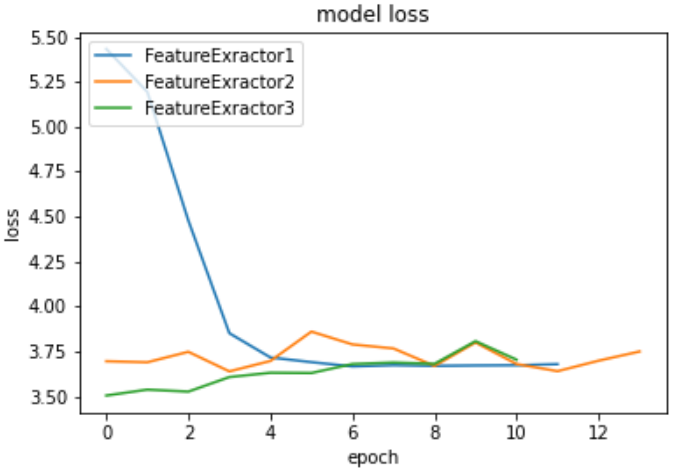


FIGURE 4.6: Loss Vs Epochs of the three Vanilla Architectures.

Chapter 5

Conclusion

5.1 Reflection

Throughout the project, I encountered several challenges, which may have been partly due to my relative inexperience with building models on this scale and the duration of the MSc. Project. However, the complexities and difficulties of the task cannot be left unacknowledged. After reflecting upon the work done, I also discovered several inadequacies. This section highlights the limitations of the work and the challenges encountered.

5.1.1 Problem formulation and baseline identification

Research attempting to achieve state-of-the-art student emotion detection is hampered by the lack of widely accepted or validated psychological models of student mental states. I realised that in its present form, e-learning settings are such an ill-formed domain with no consensus on the best student psychological model, leaving the interpretation of baseline results and metrics for success or failure as a grey area. What constitutes student affective states often varies within the research community. Therefore, finding a dataset representing the students' mental states most closely was a challenge.

This project's main objective was to demonstrate the potential of a new approach for e-learning emotion recognition which the research community could further explore. As a result, I did not find it feasible to re-implement any of the baseline methods trained for four-state student mental state classification given the duration of the project. However, this proved disadvantageous, as most baseline studies reported only the accuracy of their models without mentioning the precision. Accuracy can be misleading when there is data imbalance and gives no information about the generalisation in the predictions. As a result, it was challenging to establish that my models perform comparably and, at times,

better than current state-of-the-art models. In many cases, the F1 scores of the proposed models improved, albeit it cost a slight drop in accuracy.

5.1.2 Limitations with DAiSEE dataset

Firstly, the videos in the dataset contain only recordings of Asian participants, with a high male-to-female ratio among them, which can cause generalisation problems when employed in real-world scenarios. Another limitation with the dataset is the ambiguity in labelling the videos with the appropriate state and level. Labelling ambiguity frequently occurred in the dataset due to lack of a clear guideline for mapping facial indicators to different affective states or engagement levels of online learners. Finally, there was a huge disparity in the distribution of states and levels in the dataset, which injected significant class bias.

5.1.3 Model Execution

For feature extraction, frames were passed to the MTCNN model, which identified the most likely facial bounding box and returned face crops, which were then fed to one of the feature extractors. Depending on the feature extractor used, this process would take between 21-26 hours. Occasionally, MTCNN would return multiple faces with the same probability and that toppled the embedding dimension. The occurrence of such cases could not be identified until the embeddings were fed into the DataLoader for sequence learning, since the HPC GPU cluster produced no interim outputs or real-time logs. The problem was resolved by employing conditional clauses for the most commonly appearing dimensions and reshaping the dimensions before saving the sequence into the NumPy file. Such issues could be identified and resolved only through iterative trial and testing, which took considerable time during the project development.

In order to reduce the likelihood of overfitting, cross-validation is employed in supervised learning to tune the hyperparameters of the final model. I could not perform K-fold cross-validation in my experiments due to a lack of time, and all metrics reported are from a single run. Therefore, I refuse to comment strongly on my models' generalisation capability. Previous studies [Knyazev et al. (2017)] have shown that fine-tuning the pre-trained models used in the feature extractor can significantly improve the performance, which hasn't been done in this work. Also, the learning rate is a crucial parameter for optimisation; however, I could not extensively investigate it in this project.

Time has been the most limiting factor in my project, due to which I could not identify the best specification for the three proposed models. However, I successfully illustrated the effects of modifying these parameters through experiments on the Inception-ResNet-V1-based feature extractor. I could not make a stark comparison between the three proposed models since they have not been tested to the same extent.

5.2 Future Directions

Key-frame extraction: To improve the quality of the dataset, future work can employ key-frame selection methods that use the Cosine similarity thresholds to significantly reduce the redundant information in videos by extracting only the frames containing the most crucial information for classification, as shown by [Khenkar and Jarraya (2022)]. This also reduces the amount of data to be processed and the computational costs.

Correlation between labels: To find the important features among the 128/512-dimensional feature embeddings, correlation analysis can be performed between the features and the mental state labels as done by [Thomas and Jayagopi (2017)]. Their results show that selecting the most significant features and reducing the embedding size significantly improved the baseline results. They neglected the features with high p-value as these are statistically insignificant.

Handling Data imbalance: To address data imbalance problems, the loss function can be amended to penalise incorrect predictions on smaller classes. [Abedi and Khan (2021)] employed weighted sampling and weighted loss in their ResNet+TCN hybrid architecture and got an improvement in the detection of minority levels. Additionally, data augmentation can be used to increase the amount of data for infrequent categories.

Multi-modal data collection: Future research should be able to analyze audio signals, body gestures, eye gaze, screen interactions, and so on. Research in e-learning cannot reach its full potential without such multimodal data. Therefore, a comprehensive multimodal data collection is required. Researchers in affective computing have demonstrated that multimodal data can provide additional information during training [Dhall et al. (2018)]. Future research could examine how effective it is in multi-task student mental state recognition.

Detection of minority levels: Future research could give more emphasis on detecting the minority levels of student affective states possibly by applying anomaly detection approaches based on auto-encoders and generative adversarial networks.

5.3 Conclusion

In this work, I propose an end-to-end pipeline for "Multi-task Learning of Student Affect Recognition from Facial Expressions in-the-wild". The contribution of this work is three-fold:

1. A Proof-of-Concept(POC) for multi-task video classification of student mental States. The proposed approach predicts the level (very low, low, high and very

high) of four student mental states - Engagement, Confusion, Boredom, Frustration.

2. Spatio-Temporal analysis of facial features for student affect recognition by combining deep Convolutional Neural Network (CNNs) based feature extractors and sequence learning via LSTMs

3. A study on the performance of three SOTA networks as a feature extractor for online learning problems. The three networks are:

   - **Feature Extractor 1**: Inspired by a teacher network proposed by [Schoneveld et al. (2021)] trained on Google FEC dataset [Vemulapalli and Agarwala (2019)] and AffectNet dataset [Mollahosseini et al. (2017)]
   - **Feature Extractor 2**: ResNet18 [He et al. (2016)] pre-trained on ImageNet dataset [Deng et al. (2009)]
   - **Feature Extractor 3**: Inception-ResNet-V1 [Szegedy et al. (2015)] pre-trained on VGGFace2 [Cao et al. (2018)].

The vanilla architectures of these networks were trained on only 1429 videos due to storage constraints, contrary to 5358 train videos and 1429 validation videos used by the baseline models. Despite this limitation, all three vanilla architectures often slightly outperformed the baselines or delivered comparable results. Although the performance was not significantly greater than the state-of-the-art results, I have successfully demonstrated through Experiments 5-11 on the Inception-Resnet-V1-based model that the proposed architectures have a great potential to outperform the baselines substantially. This project's overarching objective was to demonstrate the potential of a new approaches for e-learning emotion recognition which the research community could further explore and highlight a few future directions that could be explored, which has been successfully fulfilled.

Appendix A

Design Archive

This section decribes the folders and files present in the design archive.

The design archive contains four folders and two files: FeatureExtractor1, FeatureExtractor2, FeatureExtractor3, SequenceLearning, job.slurm, requirements.txt

FeatureExtractor1 folder:
FECNet.py - Used for training the Network
Sequence.py - Used for feature Extraction
Seq_TN folder - Contains the NumPy sequence files for each Video.

FeatureExtractor2 folder:
resnet_feature.py - Used for feature Extraction
seq_resnet folder - Contains the NumPy sequence files for each Video.

FeatureExtractor3 folder:
facenet_feature.py - Used for feature Extraction
seq_facenet100 folder - Contains the NumPy sequence files for each Video extracted for 100 frames.
Seq_facenet folder - Contains the NumPy sequence files for each Video extracted for 300 frames.

SequenceLearning folder:
data_load.py - Used for Sequence Learning
Best Models folder - Contains the best checkpoints of each experiment.
LSTM.py - Contains the Sequence learning model class
test.py - Extracts classification report
ConfusionMatrix.py - Extracts confusion matrix
Analysis.ipynb - Contains code for Dataset analysis and frame similarity.

Bibliography

Affect (psychology), Aug 2022.

Ali Abedi and Shehroz S Khan. Improving state-of-the-art in detecting student engagement with resnet and tcn hybrid network. In *2021 18th Conference on Robots and Vision (CRV)*, pages 151–157. IEEE, 2021.

Mohamed Ben Ammar, Mahmoud Neji, Adel M Alimi, and Guy Gouardères. The affective tutoring system. *Expert Systems with Applications*, 37(4):3013–3023, 2010.

Kiavash Bahreini, Rob Nadolski, and Wim Westera. Towards multimodal emotion recognition in e-learning environments. *Interactive Learning Environments*, 24(3):590–605, 2016.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Lisa Feldman Barrett, Ralph Adolphs, Stacy Marsella, Aleix M Martinez, and Seth D Pollak. Emotional expressions reconsidered: Challenges to inferring emotion from human facial movements. *Psychological science in the public interest*, 20(1):1–68, 2019.

Antonio Bartolomé, Linda Castañeda, and Jordi Adell. Personalisation in educational technology: the absence of underlying pedagogies. *International Journal of Educational Technology in Higher Education*, 15(1):1–17, 2018.

Paul Baxter, Emily Ashurst, Robin Read, James Kennedy, and Tony Belpaeme. Robot education peers in a situated primary school study: Personalisation promotes child learning. *PloS one*, 12(5):e0178126, 2017.

Margaret M Bradley and Peter J Lang. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59, 1994.

Winslow Burleson. *Affective learning companions: Strategies for empathetic agents with real-time multimodal affective sensing to foster meta-cognitive and meta-affective approaches to learning, motivation, and perseverance*. PhD thesis, Massachusetts Institute of Technology, 2006.

Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, pages 67–74. IEEE, 2018.

R Carole and LEE Hyokyeong. Creating a pedagogical model that uses student self reports of motivation and mood to adapt its instruction. 2005.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

Cristina Conati and Heather Maclaren. Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction*, 19(3): 267–303, 2009.

Kevin Delgado, Juan Manuel Origgi, Tania Hasanpoor, Hao Yu, Danielle Allessio, Ivon Arroyo, William Lee, Margrit Betke, Beverly Woolf, and Sarah Adel Bargal. Student engagement dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3628–3636, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Abhinav Dhall, Amanjot Kaur, Roland Goecke, and Tom Gedeon. Emotiw 2018: Audio-video, student engagement and group-level affect prediction. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, pages 653–656, 2018.

Paul Ekman and Wallace V Friesen. Facial action coding system. *Environmental Psychology & Nonverbal Behavior*, 1978.

Clark Elliott, Jeff Rickel, and James Lester. Lifelike pedagogical agents and affective computing: An exploratory synthesis. In *Artificial intelligence today*, pages 195–211. Springer, 1999.

Tim Esler. Timesler/facenet-pytorch: Pretrained pytorch face detection (mtcnn) and facial recognition (inceptionresnet) models, 2020.

Ian J Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. Challenges in representation learning: A report on three machine learning contests.

In *International conference on neural information processing*, pages 117–124. Springer, 2013.

Abhay Gupta, Arjun D'Cunha, Kamal Awasthi, and Vineeth Balasubramanian. Daisee: Towards user engagement recognition in the wild. *arXiv preprint arXiv:1609.01885*, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Jennifer A Healey and Rosalind W Picard. Detecting stress during real-world driving tasks using physiological sensors. *IEEE Transactions on intelligent transportation systems*, 6(2):156–166, 2005.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

Rana el Kaliouby and Peter Robinson. Real-time inference of complex mental states from facial expressions and head gestures. In *Real-time vision for human-computer interaction*, pages 181–200. Springer, 2005.

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

Shoroog Khenkar and Salma Kammoun Jarraya. Engagement detection based on analyzing micro body gestures using 3d cnn. *CMC-COMPUTERS MATERIALS & CONTINUA*, 70(2):2655–2677, 2022.

Boris Knyazev, Roman Shvetsov, Natalia Efremova, and Artem Kuharenko. Convolutional neural networks pretrained on large face recognition datasets for emotion classification from video. *arXiv preprint arXiv:1711.04598*, 2017.

LB Krithika and Lakshmi Priya GG. Student emotion recognition system (sers) for e-learning improvement based on learner concentration metric. *Procedia Computer Science*, 85:767–776, 2016.

Jiacheng Liao, Yan Liang, and Jiahui Pan. Deep facial spatiotemporal network for engagement prediction in online learning. *Applied Intelligence*, 51(10):6609–6621, 2021.

Chuanhe Liu, Tianhao Tang, Kui Lv, and Minghao Wang. Multi-feature based emotion recognition for video clips. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, pages 630–634, 2018.

Javier Marín-Morales, Carmen Llinares, Jaime Guixeres, and Mariano Alcañiz. Emotion recognition in immersive virtual reality: From statistics to affective computing. *Sensors*, 20(18):5163, 2020.

Scott W McQuiggan, Sunyoung Lee, and James C Lester. Early prediction of student frustration. In *International Conference on Affective Computing and Intelligent Interaction*, pages 698–709. Springer, 2007.

Ali Mollahosseini, Behzad Hasani, and Mohammad H Mahoor. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1):18–31, 2017.

Michal Muszynski, Theodoros Kostoulas, Patrizia Lombardo, Thierry Pun, and Guillaume Chanel. Aesthetic highlight detection in movies based on synchronization of spectators' reactions, 2018.

May Honey Ohn, Urban D'Souza, and Khin Maung Ohn. A qualitative study on negative attitude toward electrocardiogram learning among undergraduate medical students. *Tzu-Chi Medical Journal*, 32(4):392, 2020.

Rosalind W Picard. *Affective computing*. MIT press, 2000.

Arti Ramesh, Dan Goldwasser, Bert Huang, Hal Daumé III, and Lise Getoor. Modeling learner engagement in moocs using probabilistic soft logic. In *NIPS workshop on data driven education*, volume 21, page 62, 2013.

Babette Renneberg, Katrin Heyn, Rita Gebhard, and Silke Bachmann. Facial expression of emotions in borderline personality disorder and depression. *Journal of behavior therapy and experimental psychiatry*, 36(3):183–196, 2005.

Fabien Ringeval, Andreas Sonderegger, Juergen Sauer, and Denis Lalanne. Introducing the recola multimodal corpus of remote collaborative and affective interactions. In *2013 10th IEEE international conference and workshops on automatic face and gesture recognition (FG)*, pages 1–8. IEEE, 2013.

Jennifer Sabourin, Bradford Mott, and James C Lester. Modeling learner affect with theoretically grounded dynamic bayesian networks. In *International Conference on Affective Computing and Intelligent Interaction*, pages 286–295. Springer, 2011.

David sandberg. Davidsandberg/facenet: Face recognition using tensorflow, 2017.

Liam Schoneveld, Alice Othmani, and Hazem Abdelkawy. Leveraging recent advances in deep learning for audio-visual emotion recognition. *Pattern Recognition Letters*, 146:1–7, 2021.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

Lin Shu, Jinyan Xie, Mingyue Yang, Ziyi Li, Zhenqi Li, Dan Liao, Xiangmin Xu, and Xinyi Yang. A review of emotion recognition using physiological signals. *Sensors*, 18 (7):2074, 2018.

Valerie Shute and Brendon Towle. Adaptive e-learning. In *Educational psychologist*, pages 105–114. Routledge, 2018.

Ewa Siedlecka and Thomas F Denson. Experimental methods for inducing basic emotions: A qualitative review. *Emotion Review*, 11(1):87–97, 2019.

Gaurav Singhal. Gaurav singhal, May 2020.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Chinchu Thomas and Dinesh Babu Jayagopi. Predicting student engagement in classrooms using facial behavioral cues. In *Proceedings of the 1st ACM SIGCHI international workshop on multimodal interaction for education*, pages 33–40, 2017.

Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006(146):10, 2006.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

Sik-Ho Tsang. Review: Densenet - dense convolutional network (image classification), Mar 2019.

Hemant Upadhyay, Yogesh Kamat, Shubham Phansekar, and Varsha Hole. User engagement recognition using transfer learning and multi-task classification. In *Intelligent Data Communication Technologies and Internet of Things*, pages 411–420. Springer, 2021.

Raviteja Vemulapalli and Aseem Agarwala. A compact embedding for facial expression similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5683–5692, 2019.

Angel de Vicente and Helen Pain. Informing the detection of the students' motivational state: an empirical study. In *International Conference on Intelligent Tutoring Systems*, pages 933–943. Springer, 2002.

Valentin Vielzeuf, Corentin Kervadec, Stéphane Pateux, Alexis Lechervy, and Frédéric Jurie. An occam's razor view on learning audiovisual emotion recognition with small training sets. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, pages 589–593, 2018.

Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *CVPR 2011*, pages 529–534. IEEE, 2011.

Rui Xia and Yang Liu. A multi-task learning framework for emotion recognition using 2d continuous space. *IEEE Transactions on affective computing*, 8(1):3–14, 2015.

Stefanos Zafeiriou, Dimitrios Kollias, Mihalis A Nicolaou, Athanasios Papaioannou, Guoying Zhao, and Irene Kotsia. Aff-wild: valence and arousal'in-the-wild'challenge. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 34–41, 2017.

Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE signal processing letters*, 23(10):1499–1503, 2016.

Zihang Zhang and Jianping Gu. Facial affect recognition in the wild using multi-task learning convolutional network. *arXiv preprint arXiv:2002.00606*, 2020.