YEDITEPE UNIVERSITY
FACULTY OF ENGINEERING

Facial Expression Recognition Using Vision Transformers and Convolutional Neural Networks

Muhammet Hakan Taştan

GRADUATION PROJECT REPORT
Department of Electrical and Electronics Engineering

Supervisor
Prof. Dr. Duygun Erol Barkana

ISTANBUL, 2024

YEDITEPE UNIVERSITY
FACULTY OF ENGINEERING

Facial Expression Recognition Using Vision Transformers and Convolutional Neural Networks

by

Muhammet Hakan Taştan

February 01, 2024, Istanbul

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

YEDITEPE UNIVERSITY

Signature of Author(s) ……

Department of Electrical and Electronics Engineering

Certified By ……

Project Supervisor, Department of Electrical and Electronics Engineering

Accepted By ……

Head of the Department of Electrical and Electronics Engineering

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my advisor, Prof. Dr. Duygun Erol Barkana, for her support and guidance throughout the project. I am truly thankful for her understanding.

I am also thankful to the faculty members and teaching assistants who have contributed to my education during my degree.

My thanks go out to my friends for their support, and encouragement. I feel fortunate to have such wonderful friends by my side.

I am deeply grateful to my parents for their unwavering support, love, and encouragement. I hope to continue making them proud in the future. Their belief in me has been a driving force throughout my life, and I am grateful for their faith in me. Thank you for always being there for me no matter what.

Lastly, I would like to express my heartfelt thanks to everyone in my life who has had an influence on me, whether big or small, for helping and shaping me into the person I am today. I appreciate the opportunities and experiences that have come my way, and I am committed to making the most of them.

Thank you all.

February, 2024

Muhammet Hakan Taştan

TABLE OF CONTENTS

ABSTRACT

Facial expression recognition (FER) is a subtask of emotion recognition that focuses on categorizing human expressions from face images. It has numerous practical applications, including security, advertising, healthcare, and recommendation systems. Recent advancements in deep learning have led to significant progress in the field. This project tries to find a lightweight and efficient method that combines convolutional neural networks (CNNs) and transformers to achieve high performance for the FER task. The proposed approach tries to improve on vision transformer on by combining a backbone network as a feature extractor to obtain fine-level features from images. Various variants were tested on the AffectNet database to find an effective approach, resulting in %55.54 accuracy on 8 class with using only 5.67M parameters.

LIST OF SYMBOLS

$*$**:** Convolution Operator

σ: Sigmoid Function

ABBREVIATIONS

FER: Facial Expression Recognition

FACS: Facial Action Coding System

AU: Action Unit

ML: Machine Learning

DL: Deep Learning

CNN: Convolutional Neural Networks

NLP: Natural Language Processing

ViT: Vision Transformer

MFN: Mixed Feature Network

CCT: Compact Convolutional Transformer

KVKK: Kişisel Verileri Koruma Kurumu

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

As we move toward the future, computers are becoming increasingly integrated into our daily lives. This growing dependence on computer systems also leads to an increase in human-computer interactions, making it necessary to improve such interactions. One area that falls under this category is emotion recognition, which involves disciplines such as psychology, neuroscience, computer sciences, and artificial intelligence. Emotion recognition has numerous applications beyond human-computer interactions, including security, advertising, healthcare, and recommendation systems [1]. Consequently, it is becoming an increasingly important research area.

Facial expression recognition (FER) is a subtask of emotion recognition that focuses on categorizing human expressions from face images. Facial expressions are one of the most potent and innate methods for humans to express their emotions and intentions. FER is a rapidly growing research area with numerous practical applications [1]. The exponential growth of image data and technological advancements has allowed new possibilities in the field, which is the reason behind the growth of this research area. Consequently, it has been the subject of numerous studies due to its many and diverse practical applications.

There are various models to represent facial expressions. These include the categorical model, the dimensional model, the facial action coding system (FACS) [2]. Emotions are defined using a set of distinct and discrete categories in categorical model [2] [3]. In contrast, Dimensional models map emotions into a multidimensional continuous space [1]. Lastly, FACS is a model that defines all possible facial expressions producible by the human face using Action Units (AU) [2]. FACS describes the states that a face can be in, rather than focusing on expressions or emotions [2].

Publicly available databases typically only provide annotations for categorical models, resulting in most FER work is limited to predicting categorical expressions. Early databases were curated from images captured in controlled environments, also known as *in-the-lab* environments [1]. However, the limitations of controlled environment databases created a demand for systems that could handle facial expressions in natural environments. As a result, databases containing such data have become increasingly popular. These databases are curated by collecting the images from web and also known as *wild* databases or *in-the-wild*

databases [2] [1]. These databases offer more content compared to their controlled environment counterparts. Additionally, these databases exhibit greater variation, but they often suffer from low image quality or class imbalance due to their curation method and large size. The AffectNet database, which is one of the largest databases, is utilized in this project [2].

Early approaches used manually extracted features [1] [4]. They used the shape and spatial information [1]. Almost all the more recent works are uses end-to-end deep learning (DL) methods [3]. The DL works used mostly convolutional neural networks (CNNs) which employ convolutions. Following the success of transformers [5] in the natural language processing (NLP) domain, attempts have been made to adapt them to other areas. As a result, there has been also an increase in recent FER works using it.

In this project, an approach that combines CNNs and Transformers is studied in order to achieve high performance for the FER task. This work uses the VGG19 network [6] and MFC network [7] as an encoder to obtain fine-level encodings from images. The encoder block outputs 28x28 encodings from 224x224 input images. Then, a patch extractor block creates patches to be input to the transformer layers. The patch extractor from Ngwe et al. [8] is utilized for the patch extraction process. The transformer layers in this study are based on the variations presented by Hassani et al. [9]. Various variants proposed based on backbone and the number of encoder layers.

1.1. Thesis Content

This report is organized as follows. Section 2 is the objective and goals of this project. Section 3 reviews existing databases and previous works on the topic. Section 4 discusses the design constraints, conditions, and engineering standards. Design approaches are also briefly discussed. Section 5 presents the proposed method and the reasoning the behind the design choices, its variants, and training process. Section 6 is where the results are discussed and compared. Finally, Section 7 concludes the thesis.

2. RESEARCH OBJECTIVE

The goal of this project is to develop a lightweight and efficient method for facial expression recognition (FER). The proposed approach aims to improve the performance of FER systems by incorporating a backbone network into the vision transformer. The goal is to achieve high performance while using fewer parameters, making the model more practical for real-world applications. Specifically, the project aims to investigate the effectiveness of combining CNNs and transformers for FER and evaluate different backbone networks as feature extractors. The goal is to experiment with different architectures and training techniques to improve the performance of the proposed method. The proposed method will be evaluated on the AffectNet database and compared to state-of-the-art approaches to analyse its strengths and weaknesses.

3. LITERATURE REVIEW

3.1. Emotion Modelling

Emotions impact numerous aspects of our everyday lives and are essential to interpersonal relationships. Thus, they draw attention to the need for researching emotions to gain a better understanding and consideration of them. Consequently, the necessity for an objective and consistent system arises. The researches will not be agreeable with each other unless the definitions used for emotions are well-defined. Also, meaning of emotions are obscure and there needs for a clarity and distinction to be able to study emotions scientifically. Therefore, understanding and accurately representing emotions through a computational model carries importance. It is a challenging yet essential task, especially in the context of artificial intelligence and human-computer interaction. This section explains the emotion modelling, exploring the foundational approaches and methodologies employed to capture the large spectrum of human emotions. Examining existing models and theories aims to establish a comprehensive understanding of the complexities involved in systematically representing emotions. Additionally, the discussion extends to the significance of emotion modelling within the broader framework of this project, setting the evaluation criteria for an emotion recognition model and addressing common pitfalls arising from inherent dilemmas of the problem domain. Despite there being several theories of emotion that seek to explain the nature and functioning of emotions, two predominant models emerge for representing them. These two models are the Categorical model and the Dimensional model. In categorical models, a single label is selected from a predefined set of labels, each linked to a specific emotion. Conversely, in dimensional models, emotions are represented through quantization on a multidimensional continuous scale [10].

Categorical models, as mentioned above, approach emotions as distinct classes. These models can either opt for a small and basic set of emotions or a more complex set that serves as an extension of basic emotions, specifically tailored to the problem domain. Various theories propose different numbers of basic emotions, but the most widely accepted one is the six basic emotions presented by Paul Ekman [11] [12]. These emotions are *anger*, *disgust*, *fear*, *joy*, *sadness*, and *surprise* [2] [11] [12] [1]. Another popular model is Plutchik's emotional wheel model [1]. It has eight basic emotions, by adding the two: *trust* and *anticipation* [1].

Moreover, the manner in which these elements are related to eachother contributes to another distinction. The mixture of these basic emotions or variations in emotion intensities is used to create more complex emotions [10]. These discrete emotions can be further reduced into three groups: *positive*, *negative*, and *neutral* [10]. These three are often employed to perform sentiment analysis. The Categorical model facilitates an easier comprehension of emotion categories. However, it also introduces challenges in the selection of distinct states, by leading subjects to choose one that does not fully represent their emotional state [10]. Additionally, it is constrained by a limited number of states. Nevertheless, its simplicity contribute to its popularity. They find widespread use for annotations of emotions in datasets.

Dimensional models are another popular taxonomy for emotion modeling. The emotions are defined in a specific point in a continuous space. They are defined in an either two dimensional [1] [13] space or three dimensional space [1]. The PAD model serves as an fairly popular example of three-dimensional models [1] [14]. The dimensions are *pleasure*, *arousal*, and *dominance*. Pleasure, also referred to as valence, is the intensity of the satisfaction of an emotional state [3]. Arousal indicates the level of physiological activity and alertness [3]. Lastly, dominance, also recognized as attention, is the sense of influencing the environment or being influenced by the the same environment [1] [14]. Later, Russell introduced the circumplex model, which consists of only two dimensions: *valence* and *arousal* [1] [13]. As emotions are situated around a circular area and the two dimensions, arousal and valence, are sufficient on their own to represent a vast majority of emotions, this space embed a much more nuanced representations [1] [13] [15]. The magnitudes for each dimension are used as scores to identify emotions. These continuous scale scores can help differentiate nuances that are otherwise hard to discern in categorical models. However, it is not hard to grasp. Additionally, they are not well-suited for clear choices [10]. The available datasets using this model are also fewer in number compared to categorical models. This scarcity can be attributed to the need for expert annotations, as annotation is considerably more challenging.

3.2. Databases

Initially, early databases were collected in a controlled laboratory setting, where subjects deliberately displayed the required emotions [2] [1]. This approach, also known as *in-the-lab* settings, provides a clean and high-quality database [2] [1]. They, however, often suffered

from limitations in terms of diversity [2]. It was primarily due to a limited number of participants, variations in head pose, and environmental conditions. Furthermore, these limitations would also affect the size of the databases, resulting in smaller sizes. Examples are JAFFE [16], MMI [1] [17] [18], CK [19] and its extension CK+ [20]. JAFFE [6] includes 213 images with 7 facial expressions, which were posed by 10 Japanese female models. Similarly, CK [7] and its extension CK+ [8] has 7 category posed by models. There are also datasets by capturing participants' facial responses to stimuli, such as DISFA [2] [21], and Belfast [2] [22].

Table 3.1 Overview of facial expression databases

Database	Year	Samples	Participants	Affect Modeling
JAFFE [16]	1998	219 images	10	Discrete (7 expressions)
CK [19]	2000	1,917 images	210	Discrete (7 expressions)
Multi-PIE [2]	2008	750,000 images	337	Discrete (7 expressions)
CK+ [20]	2010	593 video sequences	123	Discrete (7 expressions)
MMI [17] [18]	2010	740 images	25	Discrete (7 expressions)
DISFA [21]	2013	27 video clips	27	12 AUs
AFEW [23]	2012	330 video clips	Wild	Discrete (7 expressions)
FER2013 [24]	2013	35,887 images	Wild	Discrete (7 expressions)
EmotioNet [25]	2016	1,000,000 images	Wild	12 AUs
Aff-Wild [3]	2016	500 videos clips	Wild	Continuous (Valance and Arousal)
FER-Wild [26]	2016	24,000 images	Wild	Discrete (7 expressions)
AffectNet [2]	2017	450,000 (manually labeled)	Wild	Discrete (7 expressions)

				Continuous (Valance and Arousal)
RAF-DB [27]	2017	29,672 images	Wild	Discrete (7 expressions)
DFEW [28]	2020	12,059 video clips	Wild	Discrete (7 expressions)

The limitations mentioned above pushed a new way to build databases. In this method, facial expression images or videos collected from internet. It is refered as *in-the-wild* setting [2] [1]. Two of the earliest are AFEW [29], and FER2013 [24]. While former was collected from movies based on subtitles, latter database was automatically collected through the Google image search API. Other notable ones are EmotioNet [25], which consists of 1,000,000 images annotated using *action units* (AUs), and Aff-Wild [3], which is collected using videos from YouTube. It is one of the few datasets annotated for continuous affect modeling. FER-Wild [26] includes 24,000 images queried from the web using emotion terms on three different search engines. It is similar to the AffectNet [2], which collected over 1,000,000 images using same method. AffectNet is the largest database for dimensional affect modeling and also includes facial landmarks and discrete emotion annotations. However, only 450,000 images were manually annotated. RAF-DB [27] offers 29,672 images with manually crowd-sourced annotations. DFEW [28] is also annotated using the crowd source method. 12,059 clips have been professionally crowd-sourced and labeled with seven discrete expressions. Table 3.1 provides more information on the databases.

3.3. Facial Expression Recognition

Facial expression recognition systems can be categorized in several ways. Its implementations may utilize either images or videos, leading to a division into two groups: static-based and dynamic-based [1] [30]. In the former case, the emphasis is on capturing facial features using information solely from a single image. In contrast, latter consider the temporal relationships between consecutive frames in the facial expression sequence. These systems can also be grouped based on the duration and intensity of expressions: macro-FER and micro-FER [1]. Another categorization is based on the dimensions of facial images: 2D or 3D/4D [1].

Early works primarily relied on handcrafted features. These approaches used the shape of the face and its components, as well as spatial information [1]. Sometimes, relevant features were selected and redundant ones were removed. Ghimire and Lee [31] used 52 facial landmark points for automatic FER in facial sequences. Face landmarks are particular locations on the face, e.g. the eye corners, the tip of the nose and the corners of the mouth, and their angle. In 2001, Tian et al. [32] developed a system using multistates components. The components used in this study are specific regions or features of the face, such as lips, eyes, or furrows. Each component has a state based on its position or presence. The Local Binary Pattern (LBP) technique is widely used in face analysis. LBP is a texture descriptor used for feature extraction in images [33]. Zhao et al. applied LBP to FER and achieved significant results [33]. Various variants are also used and showed promising performance. Examples are LBP from three orthogonal planes (LBP-TOP) [33], LBP six intersection points (LBP-SIP) [34], and LBP-three mean orthogonal planes (LBP-MOP) [34], were also explored and demonstrated promising performance.

Deep learning FER systems are usually based on the use of CNNs. For the needs of the task, a suitable CNN architecture is chosen from among well-known architectures such as ResNet [35], VGGNet [6], MobileNet [36], EfficientNet [37] or Inception [38] as a backbone. Due to the need for high computing power and long training times for such architectures, the training is mostly done by using transfer learning. In transfer learning, the parameters of the chosen model, pre-trained on large datasets, are used to initialize the backbone model. Then, only a small number of selected layers are trained on the the specific dataset according to the needs of the task. Occasionally, fine-tuning the entire model is also considered. It is choosen according to the needs of the task. Following the success of transformers, there has been a growing interest in combining them with FER. Attention mechanism is used to the network focus more on useful features that are vital for forming expression. In this case, the attention mechanism is used to have the network focus more on useful features that are essential for the formation of the expression.

The robustness of CNN-based methods, combined with the availability of a wide range of backbone model architectures, has made them very popular. Matsugu et al. [39], proposed a transform-invariant solution, excelling in smile recognition. Oullet [40], trained a pre-trained AlexNet [41] model on CK+ [20] database. His work showed the positive impact of face

detection on model performance. Ding et al [42], introduced a VGG-16-based model that achieved an accuracy of 48.19% on the SFEW [43] database, the highest at the time of its publication. Hasani and Mahoor [44], presented a novel approach that combined the Inception [38] with ResNet [35], achieving high accuracy on multiple databases. Savchenko et al. [45], proposed several lightweight models, with the highest-performing model, EfficientNet-B2, closely following the state-of-the-art model. Wen et al. [46], showcased their transformer-based network, which attained over 80% precision at a 50% recall for all classes. Farzaneh and Qi [47], introduced a method that utilized a sparse reformulation of center loss, achieving an average accuracy of 80.44% on the RAF-DB database. Li et al. [48], proposed a novel idea that predicted expressions by generating the corresponding emotionless face, achieving state-of-the-art performance and outperforming a small ViT [49], model for prediction. The central aspect of their work was the two-stream approach utilized until the last stage.

In conclusion, the field of facial emotion recognition has witnessed significant advancements, particularly with the deep learning techniques. Each mentioned approach makes significant progress in improving FER systems, dealing with important aspects like robustness, efficiency, and precision. Overall, the progress made in facial emotion recognition holds great promise for developing systems that can effectively understand and respond to human emotions, leading to more natural and intuitive interactions between humans and machines.

4. DESIGN

4.1. Realistic Constraints and Conditions

Economy: The cost of components for this project includes the necessary hardware and software to train and implement the network. High-performance GPUs are essential for training the network, and the required infrastructure to utilize these GPUs can be substantial. For the purpose of this study, free computing services were used, including Google Colaboratory and Kaggle. However, it is important to note that these services may not always be suitable for all the time. Custom priced services, such as Amazon AWS or Sagemaker, Google Vertex.ai, and Microsoft Azure, are also available. While building the infrastructure is an option, it is not a common industry practice due to the high cost of hardware and maintenance. The economic contribution of this project lies in its potential to enhance facial expression recognition systems, which can have applications in various fields, including human-computer interaction, mental health, and security.

Environmental Issues: The training and inference processes of deep learning systems require significant computational resources, resulting in high energy consumption. Therefore, the environmental impact of these networks is a growing concern. This work employs a lightweight model that requires less computing and, consequently, less energy consumption compared to larger models. In addition, the energy consumption of cloud computing services used for training these networks varies depending on the provider. Therefore, selecting a cloud computing service that operates using renewable energy sources is an additional step towards reducing the carbon footprint of the model. This can help minimize the environmental impact of the training process and promote sustainable practices in deep learning research and applications.

Sustainability: The model's sustainability is theoretically limitless. However, its performance may vary depending on business needs and inference input distribution. To ensure its continued effectiveness, it is therefore essential to monitor and maintain the model. Regular monitoring and maintenance can prevent the need to retrain the model from scratch, allowing for its continued use. In addition, fine-tuning the model can allow it to be applied to similar tasks with ease.

Ethical: This project will adhere to all relevant engineering ethics, including data privacy and security. The data used in this project will be anonymised to protect the privacy of individuals. Furthermore, all applicable regulations, such as KVKK, will be followed to ensure ethical conduct throughout the project.

4.2. Cost of the Design

In this project, publicly available sources and free services were primarily utilized. The only cost incurred was a two-month subscription to Google Colaboratory to increase GPU run time during the training phase. The monthly cost of the subscription is 165.60 Turkish Liras, resulting in a total cost of 331 Turkish Liras for the project.

4.3. Engineering Standards

During the development of the project following standards and guidelines are followed: ISO/IEC TS 4213:2022, ISO/IEC 5338:2023, and ISO/IEC 5339:2024. ISO/IEC TS 4213:2022 is followed for assessment of the performance. ISO/IEC 5338:2023 is followed for life cycle processes. Lastly, the recommended practices in ISO/IEC 5339:2024 is followed through the development process.

4.4. Details of the Design

The proposed configurations in this work are derived from the Convolutional Context Transformer (CCT) model introduced by Hassani et al. [9]. It is designed as a way to train vision transformers from scratch on limited datasets. Training a transformer network requires substantial computing power, time, and data. Although AffectNet is one of the largest databases in the Facial Expression Recognition (FER) field, it is still not sufficient to train a one from scratch. Therefore, the CCT is a valuable tool in this context. In [9], input images are passed through a convolutional tokenizer to introduce inductive bias into the model. This tokenizer comprises a convolution with a number of filters equal to the embedding dimensions of the transformer, a ReLU [50] non-linearity, and a max pooling operation. Equation 4.1 represents this operation. In the proposed network configurations, it is replaced with a patch extractor from [8] in this work.

$$x_{tokenized} = Maxpool\Big(ReLU\big(Conv2D(x)\big)\Big)\ for\ x\ \in R^{HxWxC} \tag{4.1}$$

The patch extractor consists of three convolutional layers and one pooling layer. The first two are depth-wise separable convolutional layers, while the last one is a pointwise convolutional

layer. A global average pooling layer is applied at the end of the patch extractor to obtain the final patches. The input is not the image itself, but the fine-level features extracted from the image by a backbone network. For the backbone network, two different models are used: a truncated VGG19 [6], and a modified MFN [7]. The remaining components of the proposed configurations closely follow the original architecture [9]. The extracted patches are directed into the encoder layer, followed by a sequence pooling operation and a linear layer.

5. METHODS

In this section, the design and implementation of the proposed network is outlined. Figure 5.1 illustrates the overall architecture of the proposed network. This work builds on the methods proposed by Hassani et al. [9]. VGG19 [6] and MFN [7] networks are used as backbone choices. The encodings then used as input for a patch extraction block proposed in [8]. A global average pooling is applied to the patches and feed into a feedforward layer to map into the sequence length for transformer encoder block. Compact convolutional transformers (CCTs) [9] are used at the end for obtaining classification result.

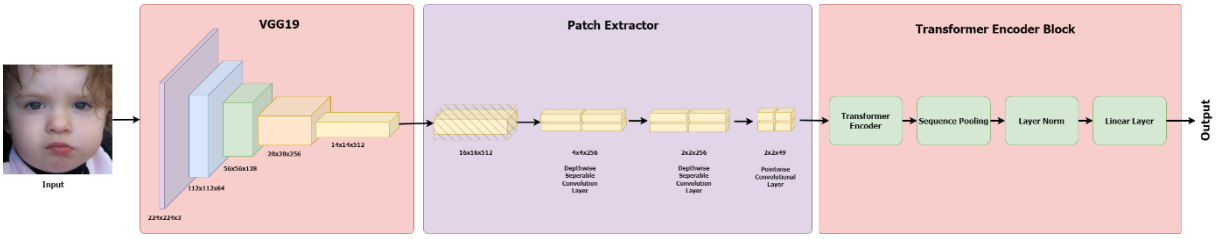

Figure 5.1 Architecture of the proposed work with VGG19 backbone

5.1. Convolutional Neural Networks

Since the success of AlexNet [41] in 2012, convolutional neural networks have become the standard for vision tasks. While CNNs became the default choice for many vision tasks after AlexNet, Kunihiko Fukushima's Neocognitron [51], proposed in 1980, laid the groundwork for CNNs by drawing inspiration from the visual systems of vertebrates [52] [53]. Similar to modern networks, Neocognitron's architecture used a filter to step through 2D input arrays. [51] [52]. LeNet-5 [54], introduced in 1998 was a successful example of combining gradient learning, convolutions, and backpropagation. After AlexNet, CNNs continued to improve with the introduction of popular architectures such as Inception [38], VGGNet [6], ResNet [35], MobileNet [36], EfficientNet [37].

Convolutional neural networks employ a convolution operator, as suggested by their name. The convolution is denoted by an asterisk and is defined as shown in Equation 2.1.

$$f * g = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \tag{5.1}$$

Images are two-dimensional and discrete, making discrete convolutions in two dimensions suitable for this purpose. These are defined as shown in Equation 2.2.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{5.2}$$

In this equation, m and n refer to the image size of m by n. The second argument is often referred to as the kernel or filter, and its size is called the kernel size. The operation is illustrated in Figure 5.2.

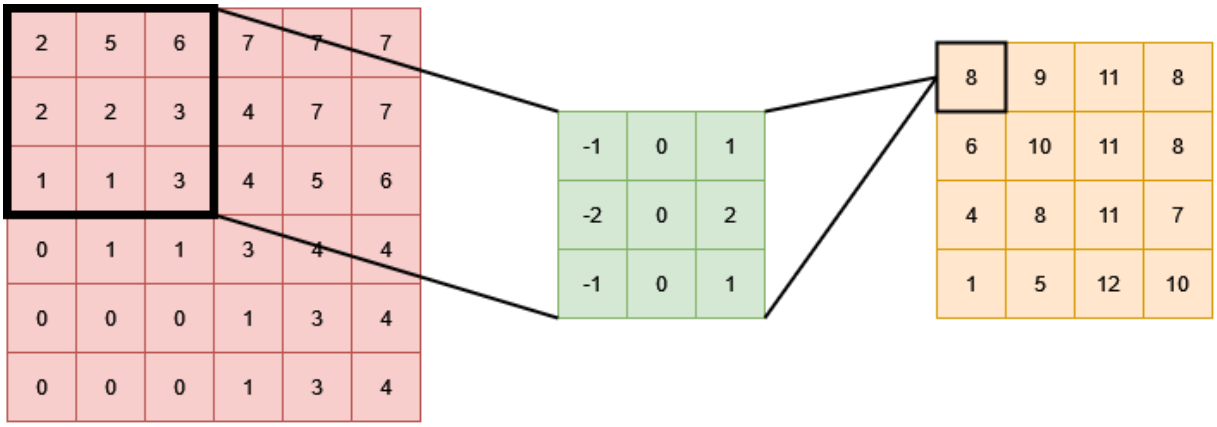

Figure 5.2 Convolution in 2D

Convolutions have inherent properties that make them perform well. They have an inductive bias that allows determining the relationship between neighbouring features of an image [9]. CNNs leverage three important concepts: sparse interactions, parameter sharing and equivariant representations [9] [53]. Every unit of the kernel is used at almost every position of the input, meaning that the same small set of parameters is used for all inputs. This is known as parameter sharing, and it reduces the number of parameters immensely. Sparse interaction occurs because of the sliding window-like operation and kernels being much smaller than the input. It reduces the number of operations needed to compute the output. Lastly, they are invariant to spatial translations. A function is equivariant to another function if it exhibits the following property, as shown in Equation 2.3

$$f\big(g(x)\big) = g(f(x)) \tag{5.3}$$

This property allows them to be unaffected by spatial transformations such as translation, rotation, flipping, or warping.

5.2. Transformers

In 2017, Vaswani et al. [5] introduced the transformer architecture for NLP tasks. Its success led to the introduction of various models based on the transformer architecture, such as BERT [55] and GPTs [56] [57]. Currently, transformers and its variants are the go-to model for all NLP tasks [58]. Subsequently, attention-based models have been adopted by a variety of tasks other than NLP, such as the Vision Transformer (ViT) [49], Swin Transformer [59], and DeiT [58].

At the heart of the architecture lies the attention mechanism. This mechanism helps the model to focus on relevant parts of the input, allowing for a better mapping. The specific category used in the architecture is called self-attention, which remains mostly the same in other variants [49] [55] [59]. In Self-attention, different parts of the input are associated to each other to compute a representation of the same input. In [5], scaled dot product is the attention function. Equation 2.4 defines it.

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5.4}$$

The function takes as input queries (Q), keys (K) of the dimension d_k and values (V) of the dimension d_v. The dot product of the queries and keys is divided by the square root of d_k. It prevents the dot product from pushing the function into parts where gradients are very small [5].

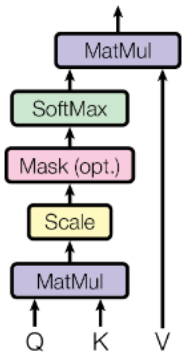

Figure 5.3 Scaled dot-product attention. (Image source: Vaswani et al. *[5]*)

In [5], the scaled dot product attention is computed multiple times in parallel, as shown in Figure 5.4. The multi-head attention improves model's ability to attend different parts of the input while maintaining its efficiency by computing them in parallel.

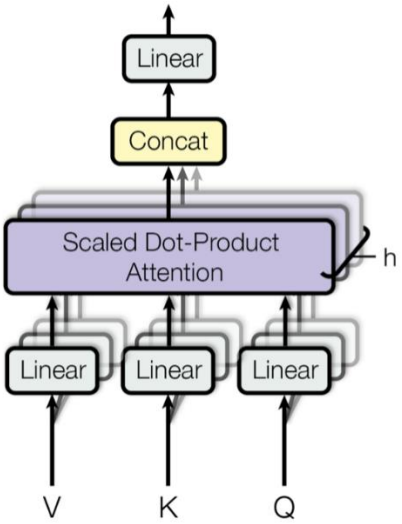

Figure 5.4 Multi-head scaled dot-product attention. (Image source: Vaswani et al. *[5]*)

The Transformers architecture comprises two main components: encoder blocks and decoder blocks. Encoder blocks contain encoder layers, while decoder blocks comprise decoder layers. Both types of layers employ multi-head attention. An encoder block consists of multi-head attention followed by a feedforward network with residual connections at the output of each sub-layers, followed by a layer normalization. The decoder block has a similar structure. The main distinction between these two is that decoder has two attention layers. However, the first one has masks in order to prevent model focusing to subsequent input. Also, some residual connections come from the encoder blocks. In Figure *5.5*, the architecture is illustrated. While the network itself is consist of encoder and decoder, both blocks do not used together all the time. Encoder only models are good for classification or recognition task, while decoder only models are used for generative models. Encoder-decoder models are suitable for sequence-to-sequence tasks such as translation.

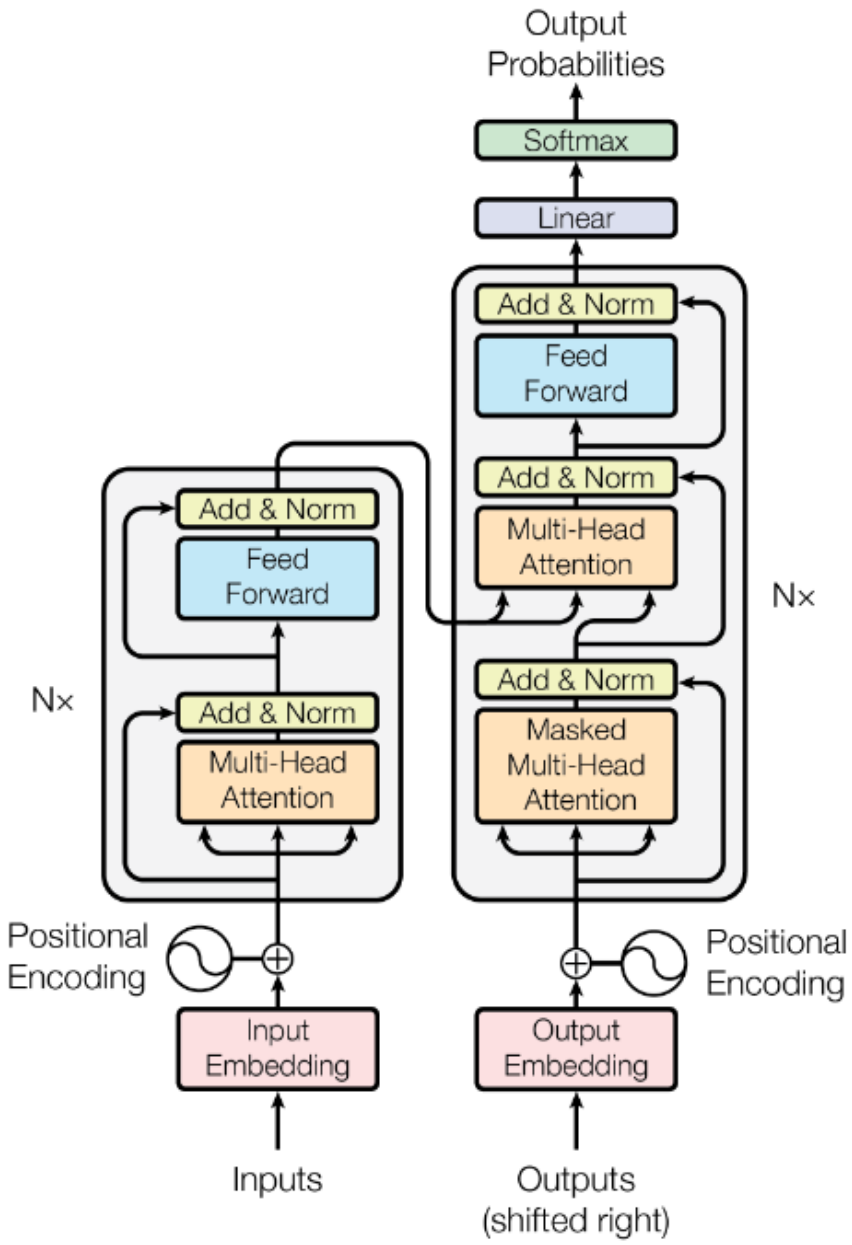

Figure 5.5 Transformer architecture. (Image source: Vaswani et al. *[5]*)

5.3. Vision Transformer and Convolutional Transformers

One of the earliest successful attempts to adapt the transformers for vision tasks is the Vision Transformation [49]. It is essentially an encoder only model, closely following the original architecture. It splits the images into flattened square patches [49]. Equation 2.5 shows the image, and Equation 2.6 shows the output of the patching process. The number of patches is calculated by dividing the product of the image width and height by the patch width and height. Vit adds a learnable class token in addition to input image. Through this token, classification is done.

$$x \in R^{HxWxC} \tag{5.5}$$

$$Patcher(x) \in R^{Nx(C*P^2)} \tag{5.6}$$

$$N = \frac{H * W}{P * P} \tag{5.7}$$

As previously mentioned, vision transformers lack the image-specific inductive bias that CNNs have. The locality and transformation invariance only come from the multilayer perceptron layer (MLP) [49]. On the other hand, attention enables the processing of information across the entire image. It is therefore much more scalable than CNNs and performs better when trained with sufficient data and time. However, this also means that training it requires a significant amount of data and computing power. Therefore, For smaller data sets and with limited computing resources, CNNs are still the better choice [9].

The introduction of DeiT [58] is the result of an attempt to reduce the its dependence on data. It is an improvement on ViT. A training paradigm called knowledge distillation [60] is used to train the model. For the distillation of a model, a teacher model is used, where the teacher model has already been trained to good performance on the task. The outputs of the both are compared for the training of the model. DeiT uses two types of distillation: soft distillation and hard distillation [58]. Former minimises the Kullback-Leibler divergence between the output of the teacher and the model [58]. It is shown in Equation 2.8 [58].

$$\mathcal{L}_{\text{global}} = (1-\lambda)\mathcal{L}_{\text{CE}}(\varphi(\text{Z}_\text{S}), \gamma) + \lambda\tau^2\text{KL}(\varphi\left(\frac{\text{Z}_\text{S}}{\tau}\right), \varphi\left(\frac{\text{Z}_\text{T}}{\tau}\right)) \tag{5.8}$$

$$\mathcal{L}_{\text{global}}^{\text{hardDistil}} = \frac{1}{2}\mathcal{L}_{\text{CE}}(\varphi(\text{Z}_\text{S}), \gamma) + \frac{1}{2}\mathcal{L}_{\text{CE}}(\varphi(\text{Z}_\text{S}), \gamma_\text{T}) \tag{5.9}$$

As shown in Equation 2.9 [58], latter is simply the use of the predictions of the teacher as true labels. DeiT introduces a distillation token in addition to the class token and used in similar manner. It interacts with other token via self-attention while trying to reproduce the output predicted by teacher model. In [58], it is verified that there indeed is an effect in the performance.

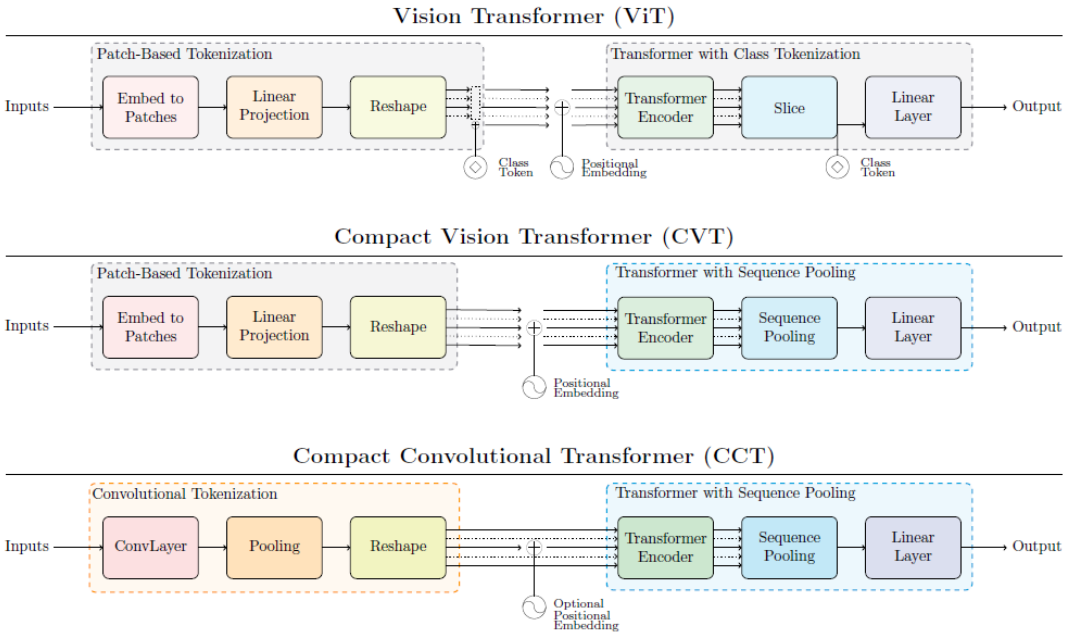

Figure 5.6 Comparison between ViT, CVT, CCT. (Image Source: Hassani et al. *[9]*)

The combination of convolutions with transformers has been another approach to improving vision transformers. These works are mainly attempts to introduce inductive bias into the vision transformers. To enable it to be trained from scratch on small data sets, Hassani et al. [9] uses CNNs. They come up with three configurations: ViT-Lite, Compact Vision Transformers (CVT), and Compact Convolutional Transformers (CCT). While ViT-Lite is a scaled down version of the original, CVT builds on ViT-Lite and introduces the sequence pooling method. Finally, CCT uses a convolutional tokenizer in addition to the improvements of CVT. Sequence pooling is the pooling of the output sequence on the based-on the attention. It replaces the class token and reduces the size of the parameters and thus the computational complexity, while preserving the relevant information over the whole sequence. CCT introduces the inductive bias by replacing the patch embedding in the original ViT with the addition of a convolutional tokenizer to the architecture. This allows the model to retain spatial information. It also allows the model, by not being bound to the patch embeddings, to be more flexible in terms of input size. Figure *5.6*, taken from [9], illustrates the differences.

5.4. VGG19

Unlike the CCT [9] where the images are directly feed into tokenizer, the input of the patch extracter comes from a truncated pre-trained VGG19 [6] in this work. To extract features from facial images, a truncated version of the VGG19 network is utilized as a backbone for this work.

VGG19 is a type of CNN that has been widely used in various computer vision tasks, including object detection and image classification. It consists of 19 weighted layers, including 16 convolutional layers, 3 fully connected layers, 5 max pooling layers, and a softmax layer for classification [6].

In this implementation, a truncated version of the network, which consisted of only the layers up to the 16th layer, which is the 4th max pooling layer. This truncated network has a smaller number of parameters than the full network, making it more computationally efficient. It has several advantages for feature extraction. Firstly, it has a simple and uniform architecture, making it easy to implement and modify. Secondly, it has been pre-trained on large-scale datasets, such as ImageNet, which shows its ability to capture high-level features that are relevant for various computer vision tasks. Finally, it has a small receptive field, which makes it well-suited for capturing fine-grained details in facial images.

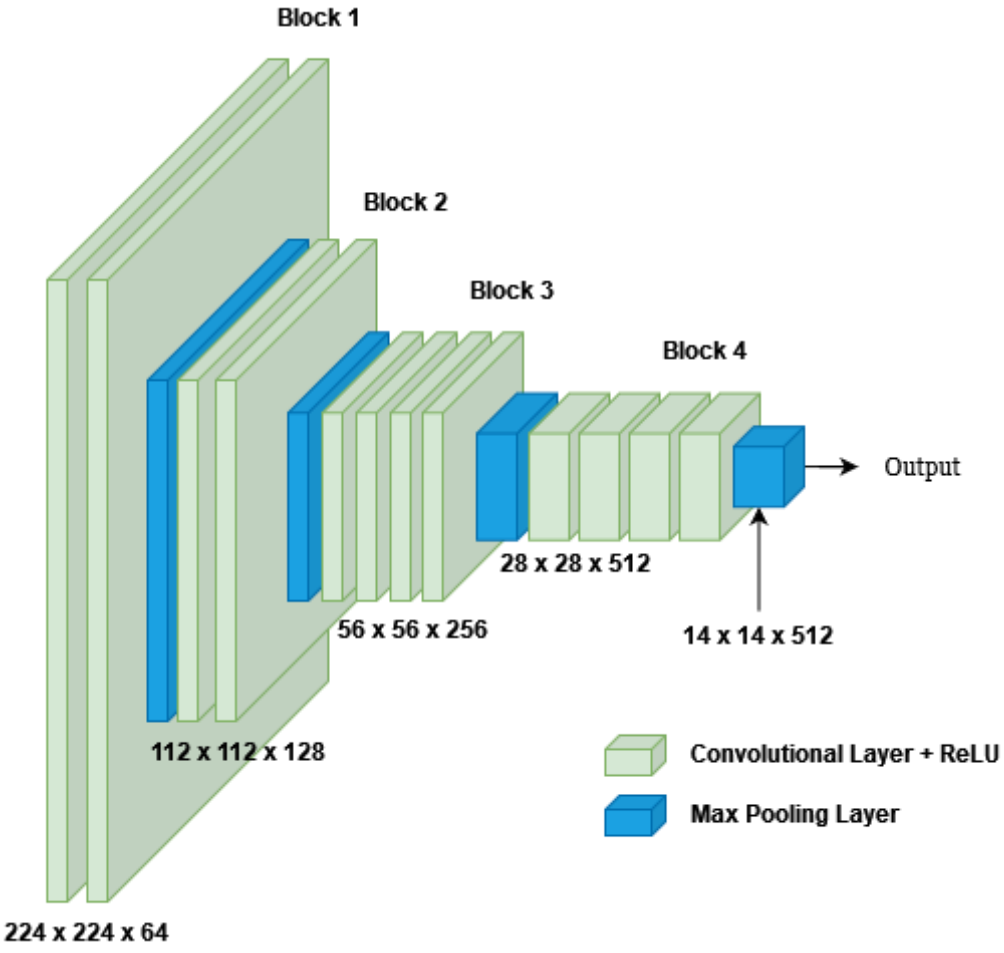

Figure 5.7 Truncated VGG19

In this implementation, extracted features from the output of the network has a spatial resolution of 14x14 and a depth of 512. The network is initialized with the pre-trained weights and freezed during the training procedure.

5.5. Mixed Feature Network (MFN)

Mixed Feature Network (MFN) is employed for the same purpose as VGG19, but it is a lightweight network and is specifically designed for extracting information from various kernel sizes [7]. It is designed to be particularly suitable for face verification tasks because popular backbones such as VGG [6] and RNN [35] are designed for larger datasets and may extract redundant information from images, which makes them prone to overfitting in relatively smaller datasets [7]. It employs two different bottleneck blocks: one residual and another non-residual. Figure 5.8 illustrates the structure of the two bottleneck blocks. The only difference between the blocks, except for the presence of the residual connection, is the stride value of the mixed depth-wise convolution block. Non-residual blocks use a stride value of two and halve the width and height of the input.

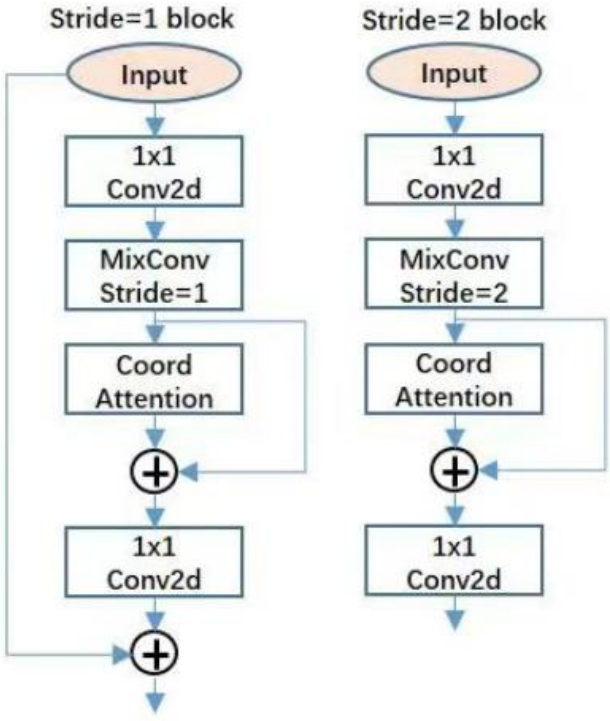

Figure 5.8 Residual block (left) and non-residual block (right). (Image Source: Zhang et al. *[7]*)

The basic convolution blocks (Conv) consist of a convolutional layer with a kernel size of 3, followed by a batch norm layer and a PReLU activation function. In contrast, mixed

convolutional blocks (Mix Depth-wise) comprise multiple basic convolutions with different kernel sizes, unlike the basic conv block. MFN also employs a coordinate attention mechanism that embeds position into channel attention, as first proposed in [61].

$$z^H = GAP(x) \tag{5.10}$$

$$z^W = GAP(x) \tag{5.11}$$

$$f = \delta(BN(conv_{1x1}([z^H; z^W]) \tag{5.12}$$

$$f^H, f^W = Split(f) \tag{5.13}$$

$$g^H = \sigma(conv_{1x1}(f^H)) \tag{5.14}$$

$$g^W = \sigma(conv_{1x1}(f^W)) \tag{5.15}$$

$$y_C(i,j) = x_C(i,j) * {g^H}_C(i) * {g^W}_C(j) \tag{5.16}$$

First, two different global pooling operations are applied to the input to obtain encodings horizontally and vertically (Equations 5.1 and 5.2). Then, a convolutional transformation with a kernel size of 1 is applied to the concatenated encodings, followed by batch normalization and a non-linearity function (Equation 5.3). The coordinate attention mechanism splits the resulting tensor into two tensors to obtain attention vectors with the same number of channels for both the horizontal and vertical coordinates of the input tensor x (Equation 5.4). Another 1x1 convolution is then applied, followed by a sigmoid function (Equations 5.5 and 5.6). The outputs g^H and g^W expanded and used as attention weights (Equation 5.7).

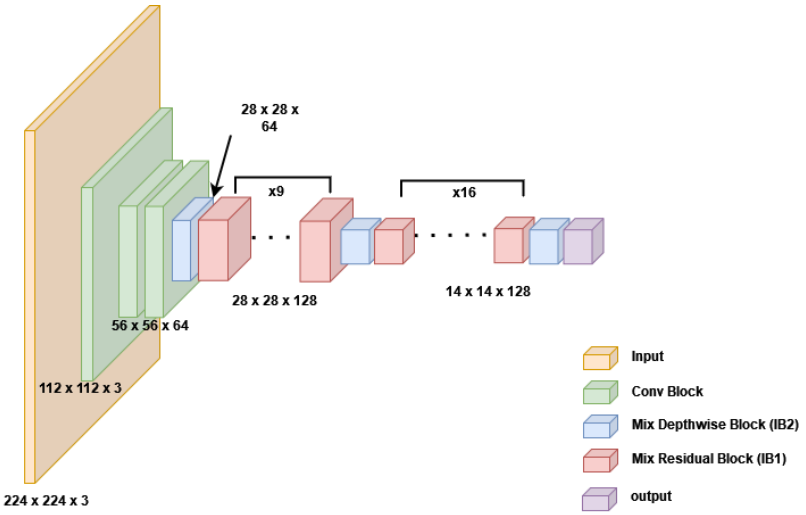

Figure 5.9 MFN Structure. "IB 1" refers to the residual bottleneck block and "IB 2" refers to the non-residual bottleneck block. n refers to the number of repetition. Input dimensions of each block is given under the block.

The original MFN architecture is closely followed except a few modifications. These modifications can be seen in Figure 5.9. While the original network accepts inputs with a resolution of 112x112, a new basic convolution block is added to change the input image size to 224x224. The other two modifications are the removal of the blocks that the size is reduced to 7x7 and the change in the output channel size from 256 to 512.

5.6. Patch Extractor

After extracting features from the backbone network, a patch extractor is applied to extract relevant features and form patches. The patch extractor proposed by Ngwe et al. [8] is utilized for this task. It is consisted of two depthwise separable convolutions and one pointwise convolution, which allowed to extract finer-grained features from facial patches. The architecture of the patch extractor is shown in Figure *5.1*. The input to the patch extractor is a 16 x 16 x 512, which is the padded output of the backbone network. Firstly a depthwise separable convolution is applied to divide into 4 patches with a kernel size of 4 x 4 and stride of 4. Next another depthwise separable convolution block is applied to to further reduce the dimension to 2 x 2 x 256. Last layer, the pointwise convoloution is applied.

5.7. Encoder

In the proposed model, a standard encoder layer is used to extract features from the input facial expressions. The input to the encoder is a 2x2x49 tensor, which is the output of the patch extractor. Embedding dimension is set to 196 and 8 attention heads are used.

5.8. Sequence Pooling

Sequence pooling enables the network to concentrate on specific segments of the input sequence after the encoder has processed it. This is similar to focusing particular aspects of a conversation after it has concluded. This capability can lead to improved performance by allowing the model to identify more nuanced details.

$$x_L = f(x_0) \in R^{b \times n \times d} \tag{5.17}$$

$$x'_L = softmax(g(x_L)^T) \in R^{b \times 1 \times n} \tag{5.18}$$

$$z = x'_L \, x_L \ \in R^{b \times 1 \times d} \tag{5.19}$$

In Equation 5.8, x_L represents the output of an L-layer transformer encoder f, where b denotes the batch size, n represents the sequence length, and d signifies the total embedding dimension. This output is fed into a linear layer, and a softmax activation function is applied

to the resulting output, as shown in Equation 5.9. As indicated in Equation 5.10, the output is produced by multiplying the matrix x_L with the weights, which is then sent to the classifier layers.

5.9. Variants

The variants proposed in this study share a common architecture, which consists of a backbone network, a patch extractor, an encoder block, and sequence pooling. The primary differences between the variants lie in the choice of backbone network and the number of encoder layers. To distinguish between the variants, a consistent naming convention has been adopted. For example, "MCCT-2" indicates that the variant utilizes the MFN network as the backbone and has two encoder layers. The initial letter of the name signifies the backbone network (M for MFN and V for VGG19), while the number represents the number of encoder layers. A total of 12 models were trained using 7 variants and various training configurations.

5.10. AffectNet

This work utilizes the AffectNet database. Due to limitations in storage capacity and computing power, the officially offered small version of the dataset is employed. This version comprises 291,650[1] images from eight distinct categories, all of which have been manually annotated. However, it only provides training and validation sets, lacking a separate test set. Consequently, the provided validation set is reserved for testing purposes. From the training set, 500 examples are randomly sampled from each category and used for validation. Table *5.1*, presents the class distribution of the provided small version, while Table 5.2 outlines the data distributions utilized for training.

As previously mentioned, AffectNet is a database curated in wild settings, meaning it was compiled using images scraped from the web through queries. Therefore, the database has a broad range of cases. This also implies the difficulty of the database. Control groups showed a 60.7% agreement between annotators, which provides the baseline for this project.

Table 5.1 Distribution of examples for the provided dataset

Labels	Training	Validation
	Number	
Neutral	74874	500

[1] Documention states the number of images as 291,651. However, provided validation set is missing 1 image

Happy	134415	500
Sad	25459	500
Surprise	14090	500
Fear	6378	500
Disgust	3803	500
Anger	24882	500
Contempt	3750	499

Table 5.2 Train, validation, and test sets distribution for training

Labels	Training	Validation	Test
	Number		
Neutral	74374	500	500
Happy	133915	500	500
Sad	24959	500	500
Surprise	13590	500	500
Fear	5878	500	500
Disgust	3303	500	500
Anger	24382	500	500
Contempt	3250	500	499

A couple of data augmentation techniques are used in order to improve performance. Each channel normalized using the stats taken from ImageNet dataset [62]. The mean values used for normalization of red, green, and blue channels are respectively as follows:

$$\mu_R = 0.485 \tag{5.20}$$

$$\mu_G = 0.456 \tag{5.21}$$

$$\mu_B = 0.406 \tag{5.22}$$

The standard deviations are as follows:

$$\sigma_R = 0.229 \tag{5.23}$$

$$\sigma_R = 0.224 \tag{5.24}$$

$$\sigma_R = 0.225 \tag{5.25}$$

Beside normalization, RandAugment [63] applied on the fly utilizing the implementation provideed by pytorch library [64]. Default values of 2 and 9 is used for respectively N and M. Resizing were not needed as all images cropped and resized to 224 x 244 by dataset curators.

No further preprocessing performed on the training data as the used small version of the data maintained well by the curators of the dataset. On validation and test sets only mean normalization performed.

$$\omega_c = \frac{1}{N_c} \tag{5.26}$$

Dataset has a huge imbalance between classes. To address this problem and minimize the impacts of the imbalance oversampling [65] is preffered. The weights used for classes is shown in Table *5.3*. The weights are calculated using the Equation 5.17.

Table 5.3 Class weights for sampling

Class	**Weight**
Neutral	0.477
Happiness	0.265
Sadness	1.421
Surprise	2.609
Fear	6.032
Disgust	10.735
Anger	1.454
Contempt	10.91

5.11. Training Process

Training was conducted on cloud computing platforms such as Google Collaboratory and Kaggle. The availability of NVIDIA P100 GPUs, NVIDIA T4 GPUs, and NVIDIA V100 GPUs determined the specific GPU used for training. A batch size of 32 was employed for all models with the VGG19 backbone, while models utilizing the MFN backbone utilized a batch size of 128. The AdamW [66] optimizer with β1 = 0.9 and β2 = 0.999 was used for all models. A learning rate of 0.001 was applied to all models. Models with the MFN backbone were trained with and without oversampling. Each model underwent at least 10 epochs of training on the full dataset.

6. RESULTS AND DISCUSSION

As previously mentioned, 12 models using 6 different variants were trained on the training set. Seven of these models were trained using oversampling, which increased the frequency of certain training examples appearing in an epoch. training examples were sampled in a manner that increased their frequency of appearance in an epoch according to the weights displayed in Table *5.3*. This was achieved by selecting the number of samples to be twice the size of the training set using the **WeightedRandomSampler** from Pytorch [64]. Essentially, this meant that the model would see twice the amount of data in an epoch, although some images would be shown more than once. An improvement in performance was therefore expected, and this was verified by the experiments. As shown in Table *6.1*, all oversampled models showed significantly better performance compared to their non-oversampled counterparts. Moreover, the main goal of improving performance on classes with fewer samples was also fairly achieved. All models with oversampling applied to the training procedure performed significantly better on those classes. The confusion matrices for each of the 12 models are given in

Appendix B. Some models with oversampling applied achieved performance comparable to state-of-the-art models [4] [7] in certain classes, such as neutral, anger, sadness, and fear, while maintaining good generalization ability with respect to other classes. Despite this, the inability to achieve a performance closer to state-of-the-art models stems from the fact that models which oversampling was applied to in the training procedure saw some decrease in the classes overrepresented in the training data, such as happiness and neutral. For instance, the best performing model, MCCT-7, shows 47% in the neutral category, while state-of-the-art models [4] and [7] show 60% and 57%, respectively The other categories lacking are disgust, fear, and contempt, which are the categories with the least examples. This is in contrast to the behaviour of other models. One other thing to note is that, all models have overfitted and failed to show a performance good enough to be better than baseline despite all efforts. One important note is that, despite all efforts, all models exhibited overfitting and failed to show a performance good enough to surpass the baseline. A more rigorous search for optimal values for random augment hyperparameters and a Stochastic Gradient Descent (SGD) with momentum variant as optimizer might be key to achieving better performance [67] [68] [69].

Another notable finding is that VGG19 does not offer significantly better performance compared to MFN, despite having nearly three times more parameters. Both 1 encoder and 2 encoder variants exhibited similar performances, regardless of the backbone used. While the VGG19 backbone showed better performance on classes with more presence, the MFN backbone demonstrated significantly outperformed on classes with fewer examples, while still achieved comparable performance on classes with more presence, albeit not as good as the VGG backbone. This is illustrated in Figure 6.1. Furthermore, during training, variants with the VGG backbone exhibited similar progress in both the validation and training sets. In contrast, variants with the MFN backbone did not show any changes in the validation data. This is depicted in the loss and accuracy plots in Figure 6.2 and Figure 6.3. This observation suggests that the VGG backbone may scale better than the MFN backbone. Additionally, increasing the number of encoder layers did not yield the desired improvement in performance.

Table 6.1 Top-1 test set accuracy comparisons. * variants trained without oversampling

Model	Performance	#Params
VCCT-1	%53.49	11.26M
VCCT-2	%53.41	11.71M
*MCCT-1	%48.19	3.00M
MCCT-1	%53.24	3.00M
*MCCT-2	%44.16	3.44M
MCCT-2	% 54.19	3.44M
*MCCT-3	%41.49	3.89M
MCCT-3	%51.21	3.89M
*MCCT-6	% 45.26	5.23M
MCCT-6	% 52.79	5.23M
*MCCT-7	% 40.79	5.67M
MCCT-7	% 55.54	5.67M

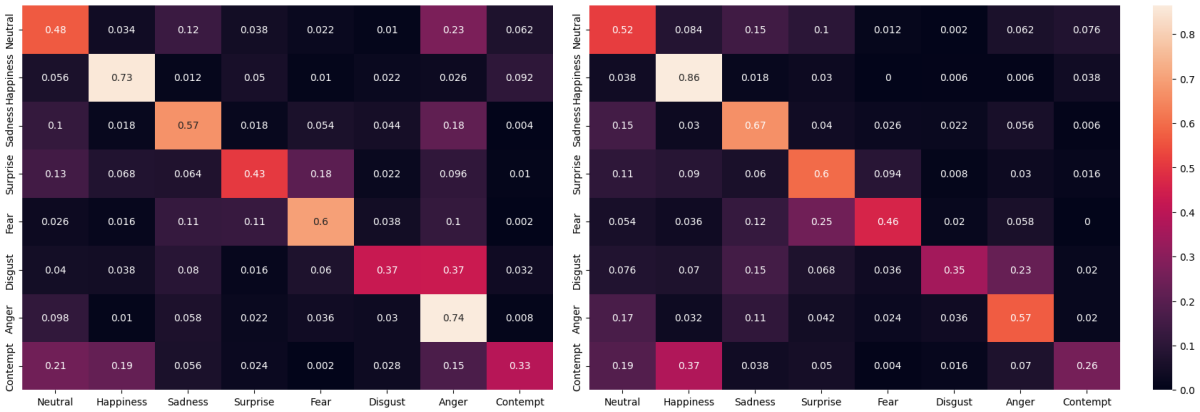

Figure 6.1 Confusion Matrices of MCCT-1 (left) and VCCT-1 (right)

Figure 6.2 MCCT-1 loss plot (top) and VCCT-1 loss plot (bottom)

Figure 6.3 MCCT-1 accuracy plot (top) and VCCT-1 accuracy plot (bottom)

7. CONCLUSION

In this study, a lightweight and efficient method that combines convolutional neural networks (CNNs) and transformers for facial expression recognition (FER) is proposed. The proposed approach aims to improve upon the vision transformer in Hassani et al. [9] by incorporating a backbone network to extract fine-level features from images. Different variants are experimented on the AffectNet database to find an effective approach, achieving considerable performance compared to the best results on AffectNet[2]. The training plots for all experiments are shared in Appendix A, and Appendix B contains the confusion matrices for all trained models. The code for the model and training will be publicly shared on GitHub.

The backbone networks used in the study did not show a considerable difference in performance, despite their size difference. However, the VGG19 network showed some indication of better scaling ability compared to MFN. Oversampling proved to enhance performance in all variants experimented with. Best performing variant MCCT-7 with oversampling applied during training procedure achieved %55.54 on officially provided AffectNet validation set on 8 classes. While performing closer to or better than [4] and [7], two of the currently best performing models, in some categories, it failed to come close to those two models in mean accuracy for all categories, with those two models achieving 63.77% and 64.25%, respectively. This mainly resulted from lower performance on over-represented classes and not performing well enough on under-represented classes, despite oversampling. Given that MCCT-1 showed better performance on 20 epochs compared to 10 epochs, even though it was overfitting, it suggests that if MCCT-7 were trained for more epochs, it would also improve and show better performance on over-represented classes, which is one of the main reasons for its failure. Moreover, switching the optimizer from AdamW to Stochastic Gradient Descent (SGD) would help to improve better performance. As studied in works [67], [68], [69], AdamW, the optimizer used in this study, might limit the model's ability to generalize. Thus, there are several directions for future work. First, explore the use of different backbone networks to further improve the efficiency of the proposed method. Second, explore the use of different optimizers. Lastly, conduct a more rigorous search for hyperparameters of random augment or explore the use of different data

[2] Please refer to the Papers With Code for benchmark list

augmentation techniques to further improve the generalization ability of the proposed method.

REFERENCES

[1] Y. Wang, W. Song, W. Tao, A. Liotta, D. Yang, X. Li, S. Gao, Y. Sun, W. Ge, W. Zhang and W. Zhang, "A systematic review on affective computing: emotion models, databases, and recent advances," *Information Fusion,* vol. 83–84, pp. 19-52, 2022.

[2] A. a. H. B. a. M. M. H. Mollahosseini, "AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild," *IEEE Transactions on Affective Computing,* vol. 10, no. 1, pp. 18-31, 2019.

[3] S. Zafeiriou, A. Papaioannou, I. Kotsia, M. Nicolaou and G. Zhao, "Facial Affect "in-the-wild": A survey and a new database," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Las Vegas, NV, USA, 2016.

[4] J. Mao, R. Xu, X. Yin, Y. Chang, B. Nie and A. Huang, "POSTER V2: A simpler and stronger facial expression recognition network," *arXiv preprint arXiv:2301.12149,* 2023.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems,* vol. 30, 2017.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.

[7] S. Zhang, Y. Zhang, Y. Zhang, Y. Wang and Z. Song, "A Dual-Direction Attention Mixed Feature Network for Facial Expression Recognition," *Electronics,* vol. 12, no. 17, p. 3595, 2023.

[8] J. L. Ngwe, K. M. Lim, C. P. Lee and T. S. Ong, "PAtt-Lite: Lightweight Patch and Attention MobileNet for Challenging Facial Expression Recognition," *arXiv preprint arXiv:2306.09626,* 2023.

[9] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li and H. Shi, "Escaping the Big Data Paradigm with Compact Transformers," *arXiv preprint arXiv:2104.05704,* 2022.

[10] P. Sreeja and G. Mahalakshmi, "Emotion models: a review," *International Journal of Control Theory and Applications,* vol. 10, no. 8, pp. 651-657, 2017.

[11] P. Ekman, "An argument for basic emotions," *Cognition and Emotion,* vol. 6, no. 3/4, pp. 169-200, 1992.

[12] P. Ekman, "Are There Basic Emotions," *Psychological Review,* vol. 99, no. 3, pp. 550-553, 1992.

[13] J. A. Russell, "A circumplex model of affect.," *Journal of personality and social psychology,* vol. 39, no. 6, p. 1161, 1980.

[14] I. Bakker, T. van der Voordt, P. Vink and J. de Boon, "Pleasure, arousal, dominance: Mehrabian and Russell revisited," *Current Psychology,* vol. 33, no. 3, pp. 405-421, 2014.

[15] J. A. Russell and G. Pratt, " A description of the affective quality attributed to environments," *Journal of Personality and Social Psychology,* vol. 38, no. 2, p. 311–322, 1980.

[16] M. Lyons, S. Akamatsu, M. Kamachi and J. Gyoba, "Coding facial expressions with Gabor wavelets," in *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, Nara, Japan, 1998.

[17] M. Pantic, M. Valstar, R. Rademake and L. Maat, "Web-based database for facial expression analysis," in *IEEE International Conference on Multimedia and Expo*, Amsterdam, Netherlands, 2005.

[18] M. F. Valstar and M. Pantic, "Induced Disgust, Happiness and Surprise: an Addition to the MMI Facial," 2010.

[19] T. Kanade, J. F. Cohn and Y. Tian, "Comprehensive Database for Facial Expression Analysis," in *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, Grenoble, France, 2000.

[20] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar and I. Matthews, "The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified

expression," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, San Francisco, CA, USA, 2010.

[21] S. M. Mavadati, M. H. Mahoor, K. Bartlett, P. Trinh and J. F. Cohn, "DISFA: A Spontaneous Facial Action Intensity Database," *IEEE Transactions on Affective Computing,* vol. 4, no. 2, pp. 151-160, 2013.

[22] I. Sneddon, M. McRorie, G. McKeown and J. Hanratty, "The Belfast Induced Natural Emotion Database," *IEEE Transactions on Affective Computing,* vol. 3, no. 1, pp. 32-41, 2012.

[23] A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Collecting Large, Richly Annotated Facial-Expression Databases from Movies," *IEEE MultiMedia,* vol. 19, no. 3, pp. 34-41, 2012.

[24] I. J. Goodfellow and e. al., "Challenges in Representation Learning: A report on three machine learning contests," *Neural Information Processing,* pp. 117-124, 2013.

[25] C. F. Benitez-Quiroz, R. Srinivasan and A. M. Martinez, "EmotioNet: An Accurate, Real-Time Algorithm for the Automatic Annotation of a Million Facial Expressions in the Wild," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016.

[26] A. Mollahosseini, B. Hassani, M. J. Salvador, H. Abdollahi, D. Chan and M. H. Mahoor, "Facial Expression Recognition from World Wild Web," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016.

[27] S. Li and W. Deng, "Reliable Crowdsourcing and Deep Locality-Preserving Learning for Unconstrained Facial Expression Recognition," *IEEE Transactions on Image Processing,* vol. 28, no. 1, pp. 356-370, 2019.

[28] X. Jiang, Y. Zong, W. Zheng, C. Tang, W. Xia, C. Lu and J. Liu, "DFEW: A Large-Scale Database for Recognizing Dynamic Facial Expressions in-the-Wild," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020.

[29] A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Collecting Large, Richly Annotated Facial-Expression Databases from Movies," *IEEE MultiMedia,* vol. 19, no. 3, pp. 34-41, 2012.

[30] S. Li and W. Deng, "Deep Facial Expression Recognition: A Survey," *IEEE Transactions on Affective Computing,* vol. 13, no. 3, pp. 1195-1215, 2022.

[31] D. Ghimire and J. Lee, "Geometric Feature-Based Facial Expression Recognition in Image Sequences Using Multi-Class AdaBoost and Support Vector Machines," *Sensors,* vol. 13, no. 6, p. 7714–7734, 2013.

[32] Y.-I. Tian, T. Kanade and J. Cohn, "Recognizing action units for facial expression analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 23, no. 2, pp. 97-115, 2001.

[33] G. Zhao and M. Pietikainen, "Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 29, no. 6, pp. 915-928, 2007.

[34] Y. Wang, J. See, R. C.-W. Phan and Y.-H. Oh, "Efficient Spatio-Temporal local binary patterns for spontaneous facial Micro-Expression recognition," *PLOS ONE,* vol. 10, no. 5, 2015.

[35] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[36] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, "MobileNets: efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861,* 2017.

[37] M. L. Q. Tan, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[38] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[39] M. Matsugu, K. Mori, Y. Mitari and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Networks,* vol. 16, no. 5-6, p. 555–559, 2003.

[40] S. Ouellet, "Real-time emotion recognition for gaming using deep convolutional network features," *arXiv preprint arXiv:1408.3750,* 2014.

[41] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, 2012.

[42] H. Ding, S. K. Zhou and R. Chellappa, "FaceNet2ExpNet: Regularizing a Deep Face Recognition Net for Expression Recognition," in *12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, Washington, DC, USA, 2017.

[43] A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, Barcelona, Spain, 2011.

[44] B. Hasani and M. H. Mahoor, "Facial expression recognition using enhanced deep 3D convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, Honolulu, HI, USA, 2017.

[45] A. V. Savchenko, L. V. Savchenko and I. Makarov, "Classifying Emotions and Engagement in Online Learning Based on a Single Facial Expression Recognition Neural Network," *IEEE Transactions on Affective Computing,* vol. 13, no. 4, pp. 2132-2143, 2022.

[46] Z. Wen, W. Lin, T. Wang and G. Xu, "Distract Your Attention: Multi-Head Cross Attention Network for Facial Expression Recognition," *Biomimetics,* vol. 8, no. 2, p. 199, 2023.

[47] A. H. Farzaneh and X. Qi, "Facial Expression Recognition in the Wild via Deep Attentive Center Loss," in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Waikoloa, HI, USA, 2021.

[48] J. Li, J. Nie, D. Guo, R. Hong and M. Wang, "Emotion Separation and Recognition from a Facial Expression by Generating the Poker Face with Vision Transformers," *arXiv preprint arXiv:2207.11081,* 2023.

[49] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, Houlsby and Neil, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929,* 2020.

[50] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375,* 2018.

[51] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics,* vol. 36, no. 4, p. 193–202, 1980.

[52] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks,* vol. 61, pp. 85-117, 2015.

[53] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[54] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, no. 11, pp. 2278-2324, 1998.

[55] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805,* 2018.

[56] A. Radford et al., "Improving language understanding by generative pre-training," *OpenAI,* 2018.

[57] L. M. a. F.-S. Learners, "Language models are few-shot learners," in *Advances in neural information processing systems*, 2020.

[58] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*, 2021.

[59] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[60] G. Hinton, O. Vinyals and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531,* 2015.

[61] Q. Hou, D. Zhou and J. Feng, "Coordinate attention for efficient mobile network design," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021.

[62] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 2009.

[63] E. D. Cubuk, B. Zoph, J. Shlens and Q. V. Le, "RandAugment: Practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020.

[64] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in neural information processing systems,* 2019.

[65] M. Buda, A. Maki and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural networks,* vol. 106, pp. 249-259, 2018.

[66] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," *arXiv preprint arXiv:1711.05101,* 2019.

[67] D. Zou, Y. Cao, Y. Li and Q. Gu, "Understanding the generalization of adam in learning neural networks with proper regularization," *arXiv preprint arXiv:2108.11371,* 2021.

[68] N. S. Keskar and R. Socher, "Improving generalization performance by switching from adam to sgd," *arXiv preprint arXiv:1712.07628,* 2017.

[69] P. Zhou, J. Feng, C. Ma, C. Xiong, S. Hoi and W. E, "Towards theoretically understanding why sgd generalizes better than adam in deep learning," *Advances in Neural Information Processing Systems,* vol. 33, pp. 21285-21296, 2020.

APPENDICES

Appendix A

Training and Validation Loss and Accuracy Plots

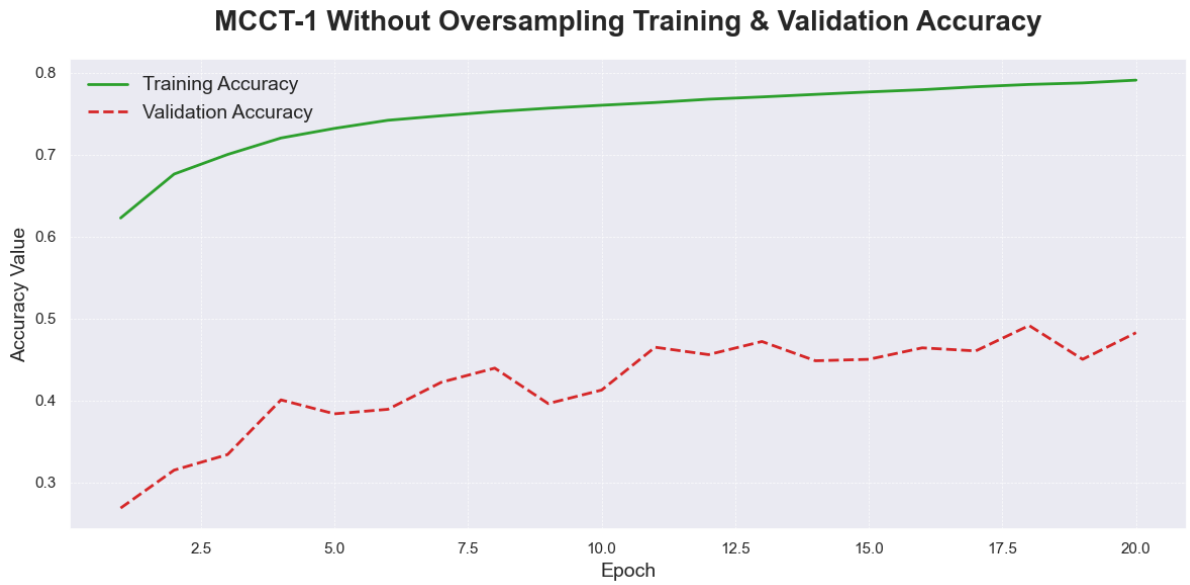

Figure 0.1 Accuracy plot for MCCT-1 without oversampling

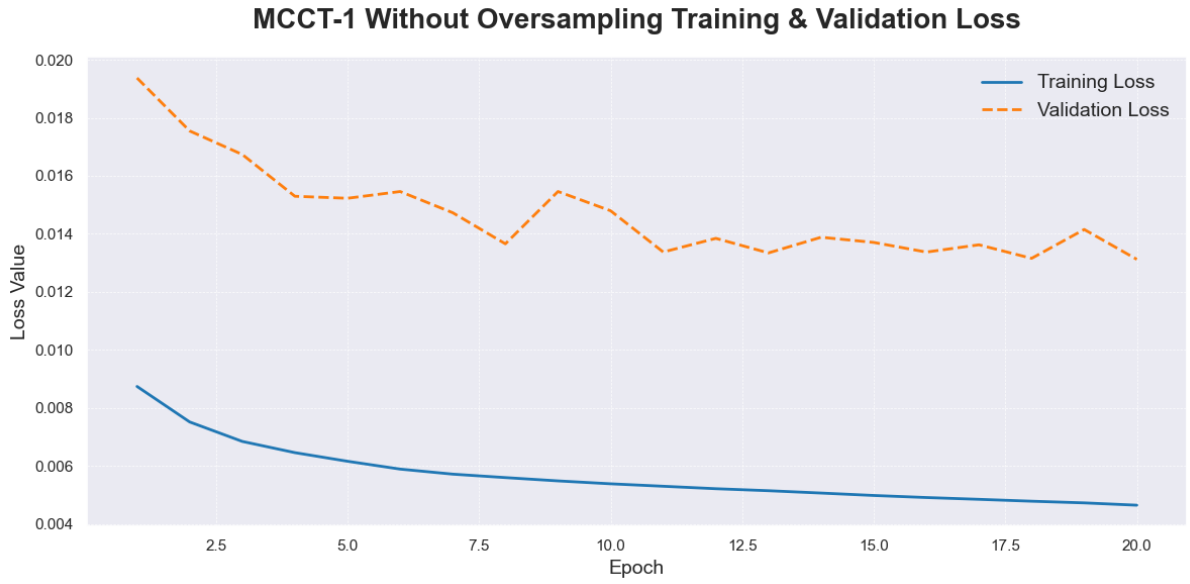

Figure 0.2 Loss plot for MCCT-1 without oversampling

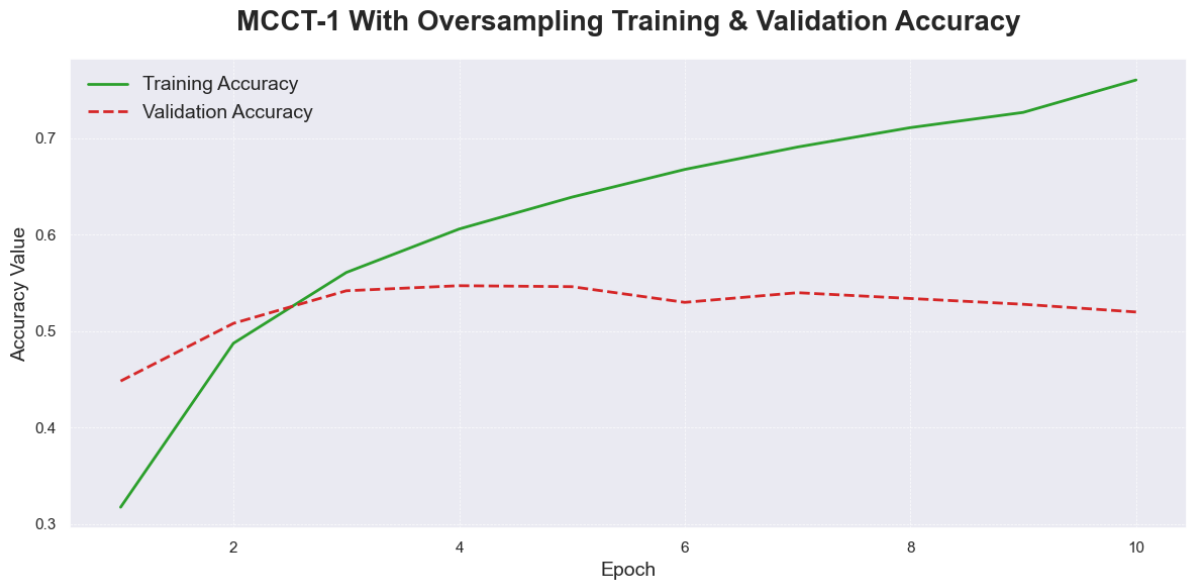

Figure 0.3 Accuracy plot for MCCT-1 with oversampling

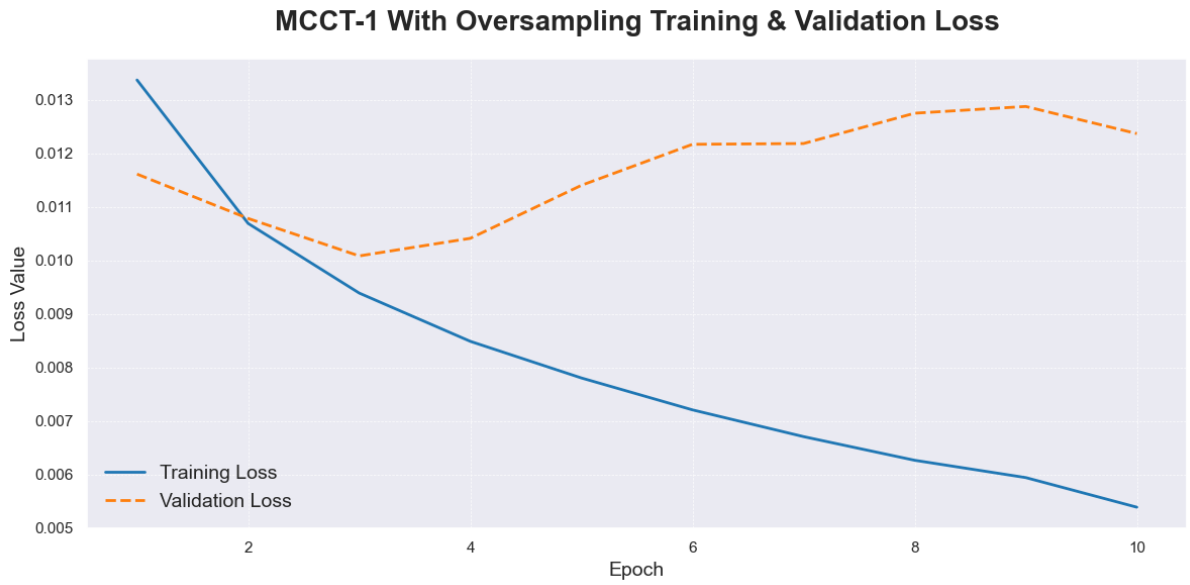

Figure 0.4 Loss plot for MCCT-1 with oversampling

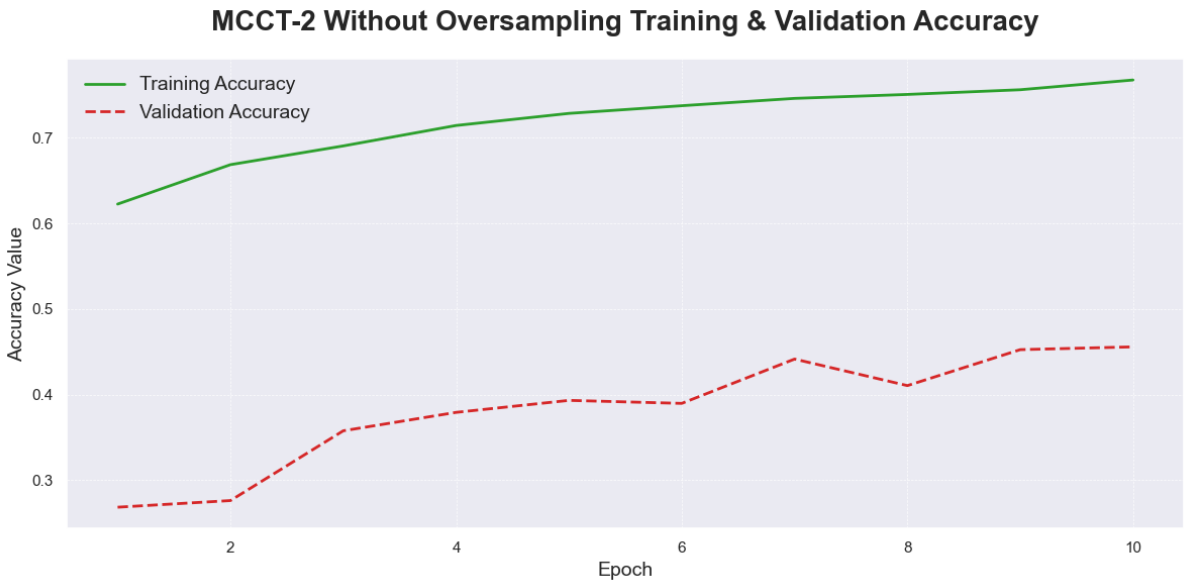

Figure 0.5 Accuracy plot for MCCT-2 without oversampling

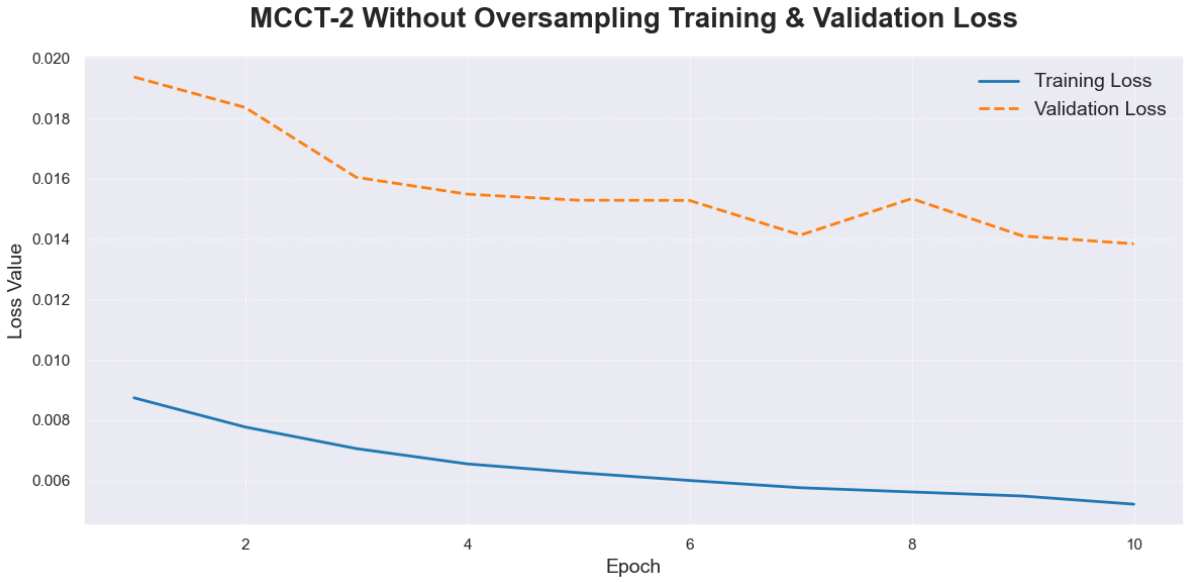

Figure 0.6 Loss plot for MCCT-2 without oversampling

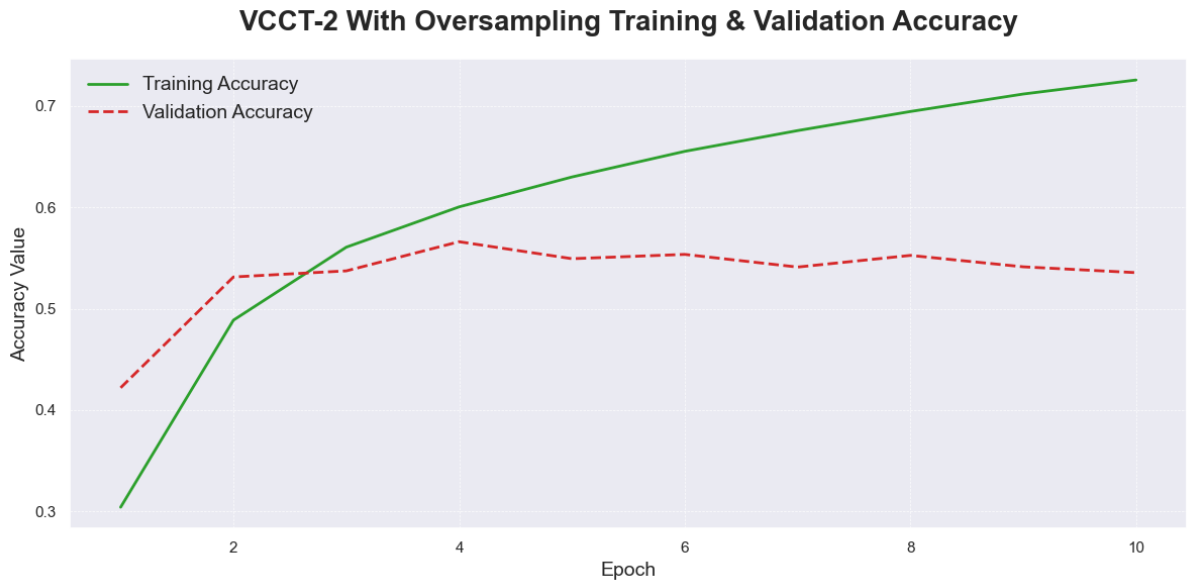

Figure 0.7 Accuracy plot for MCCT-2 with oversampling

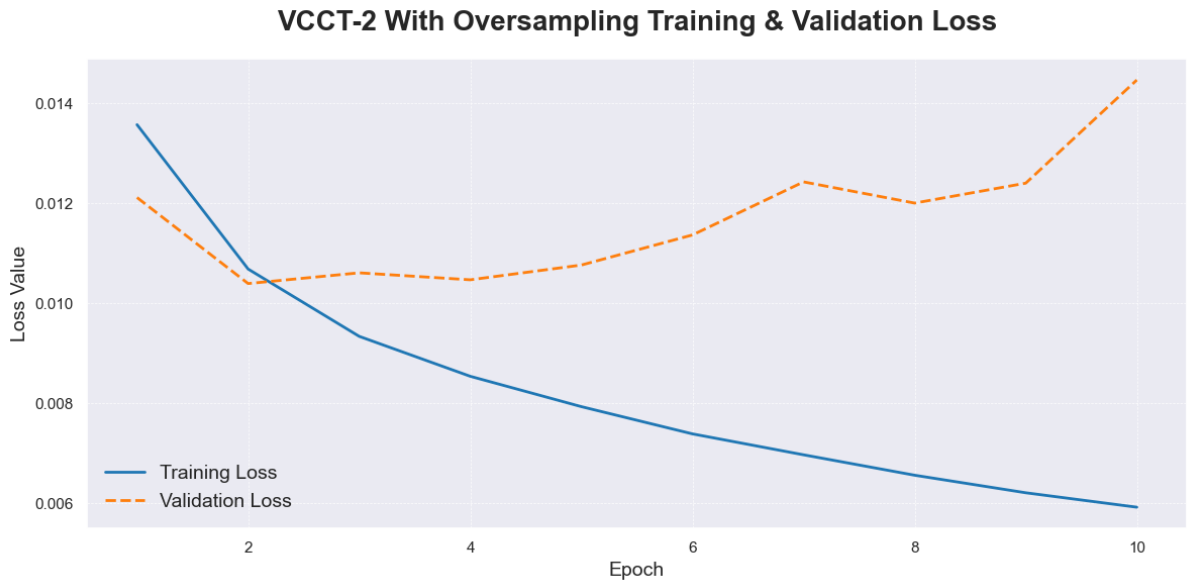

Figure 0.8 Loss plot for MCCT-2 with oversampling

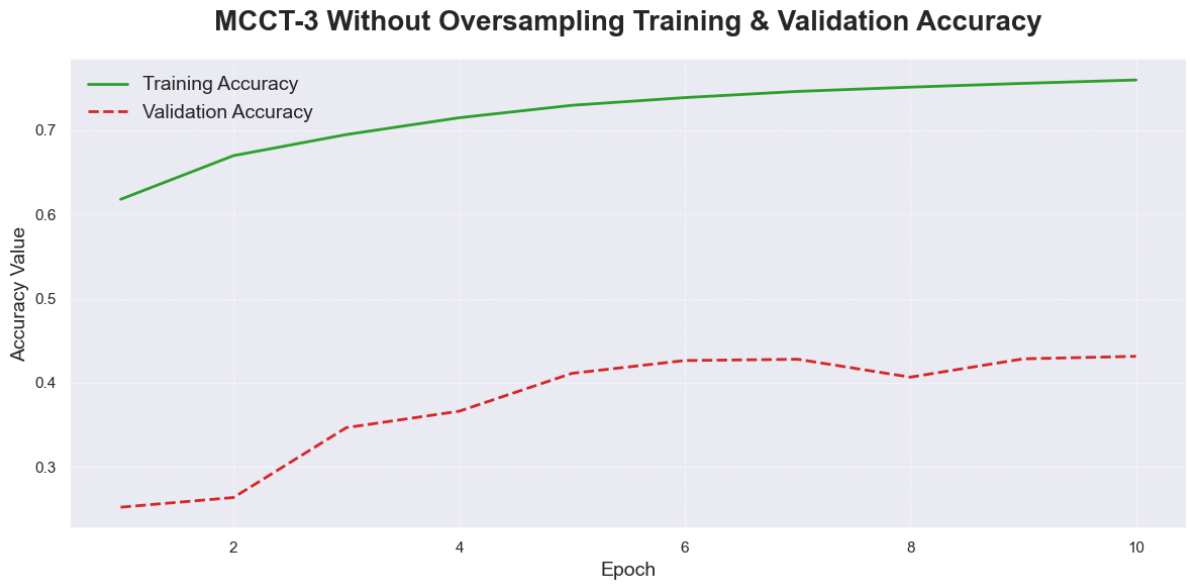

Figure 0.9 Accuracy plot for MCCT-3 without oversampling

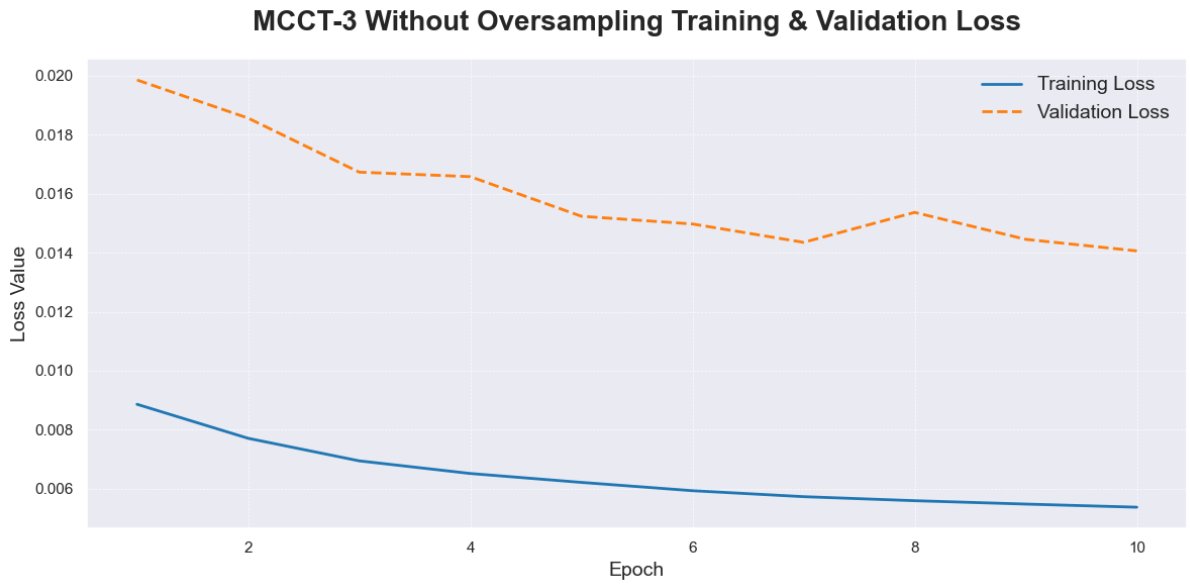

Figure 0.10 Loss plot for MCCT-3 without oversampling

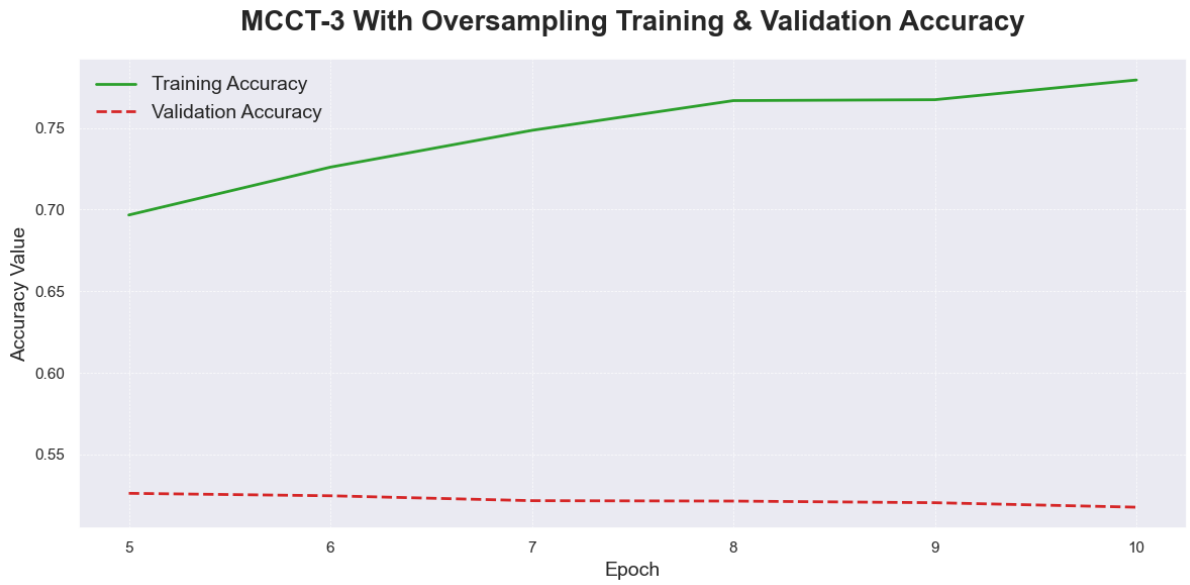

Figure 0.11 Accuracy plot for MCCT-3 with oversampling

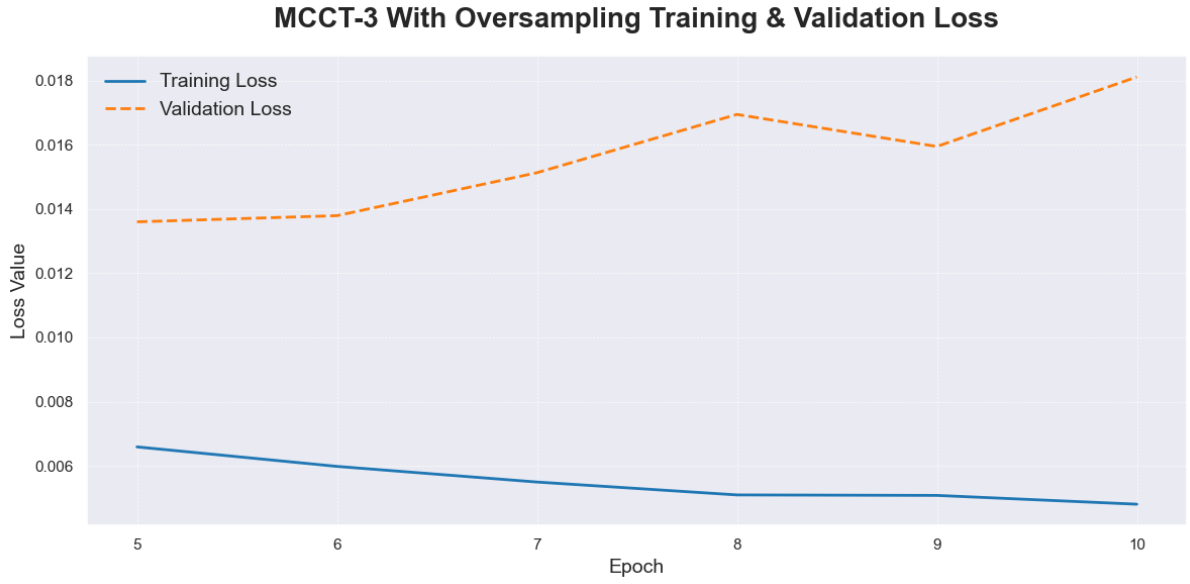

Figure 0.12 Loss plot for MCCT-3 with oversampling

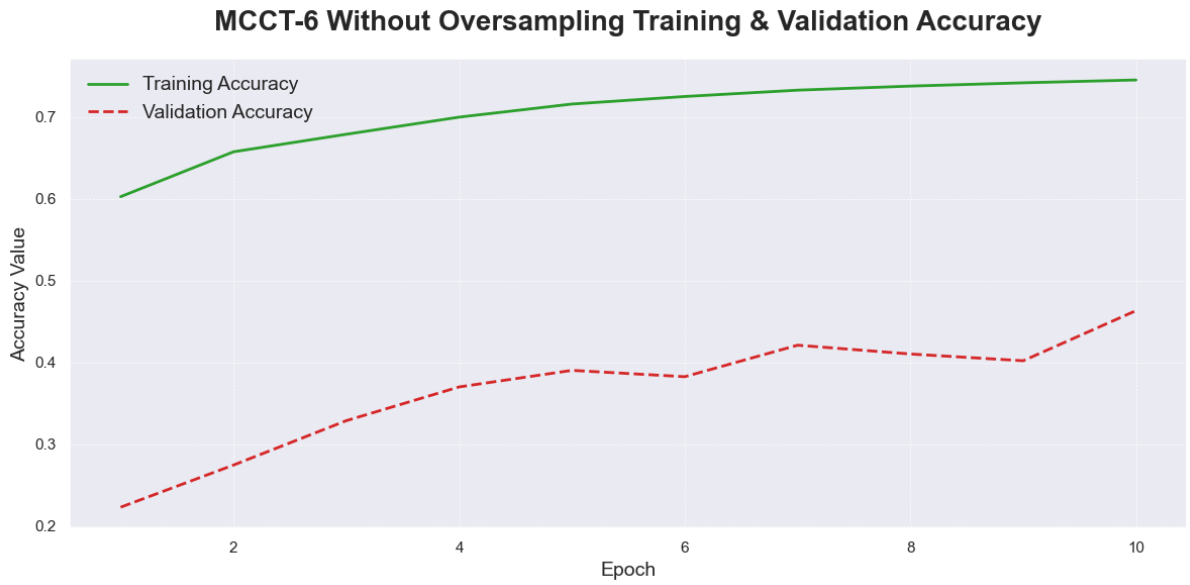

Figure 0.13 Accuracy plot for MCCT-6 without oversampling

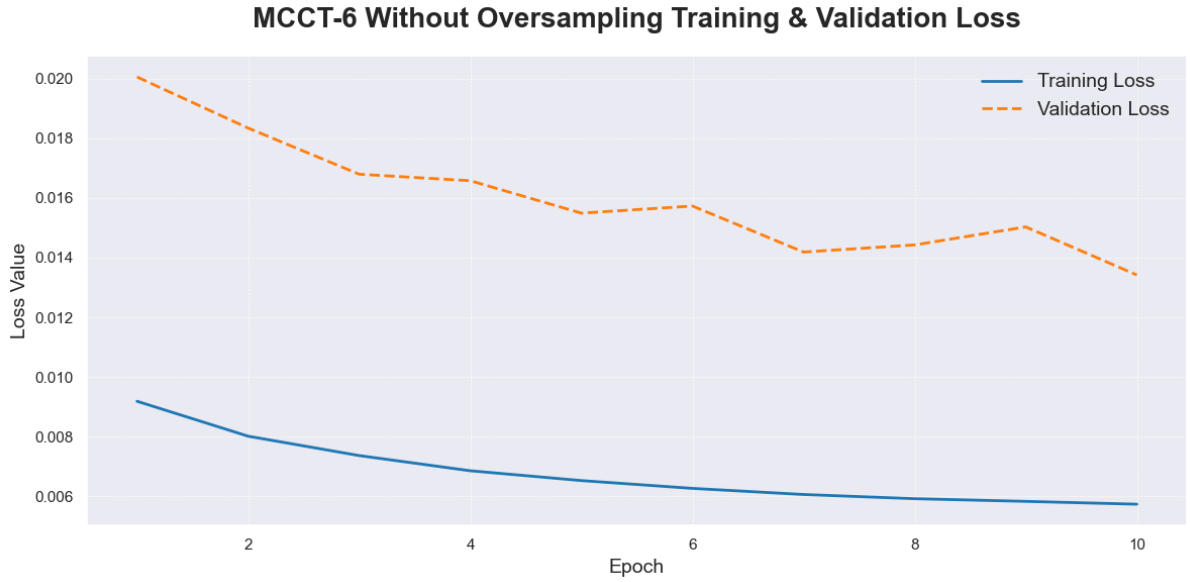

Figure 0.14 Loss plot for MCCT-6 without oversampling

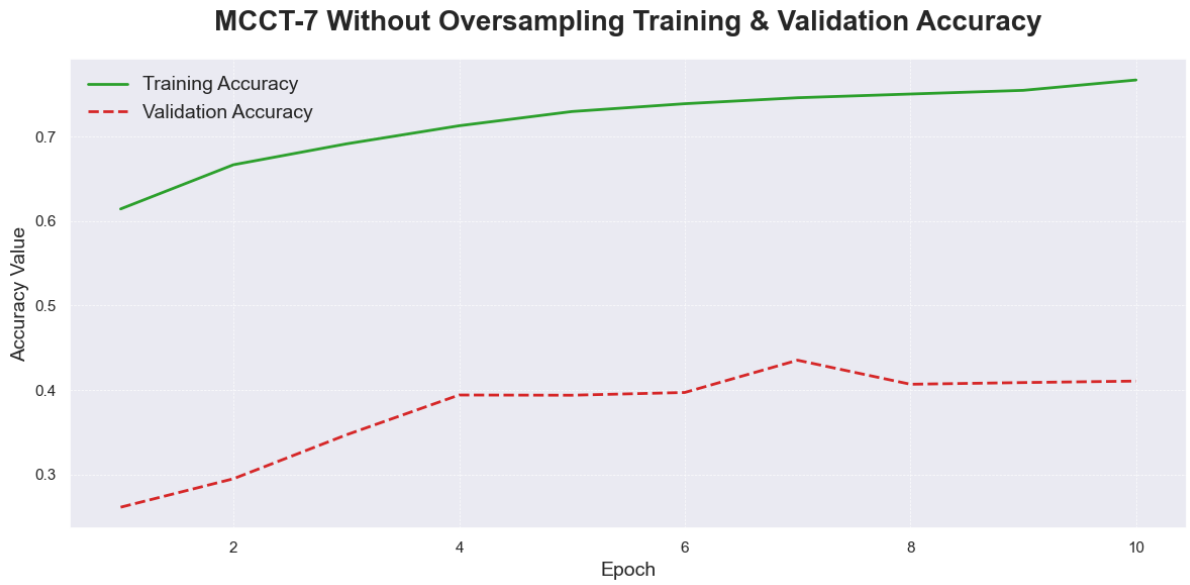

Figure 0.15 Accuracy plot for MCCT-7 without oversampling

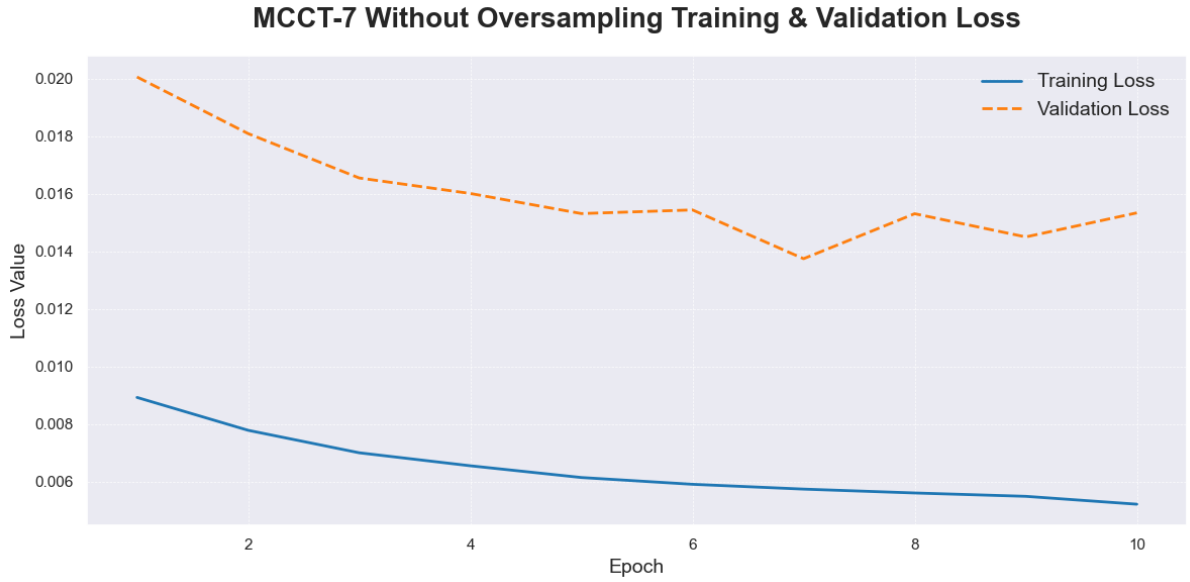

Figure 0.16 Loss plot for MCCT-7 without oversampling

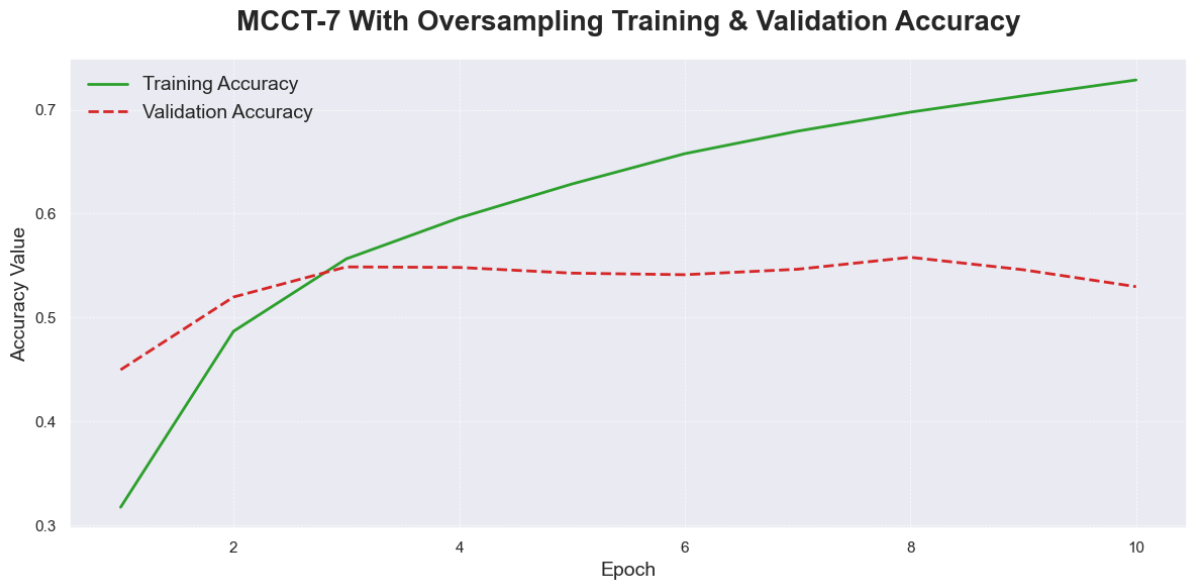

Figure 0.17 Accuracy plot for MCCT-7 with oversampling

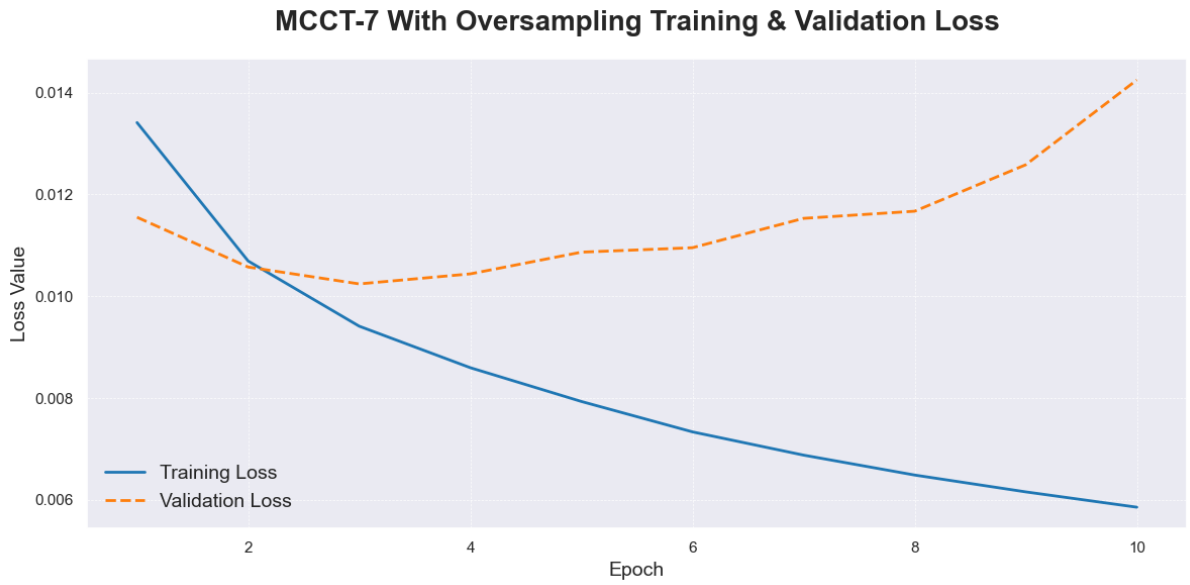

Figure 0.18 Loss plot for MCCT-7 with oversampling

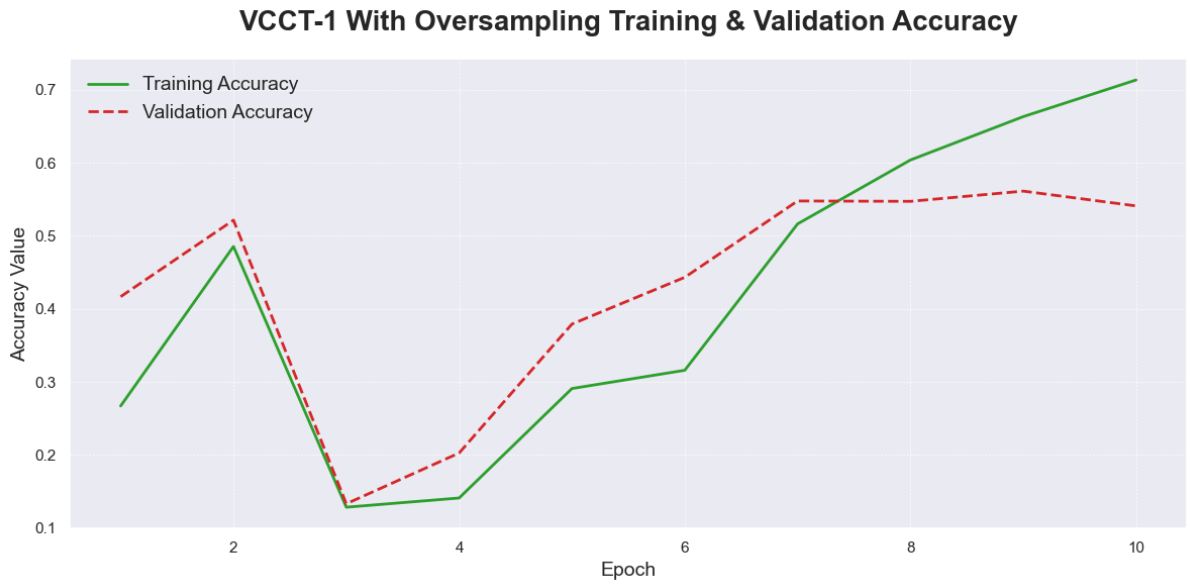

Figure 0.19 Accuracy plot for VCCT-1 with oversampling

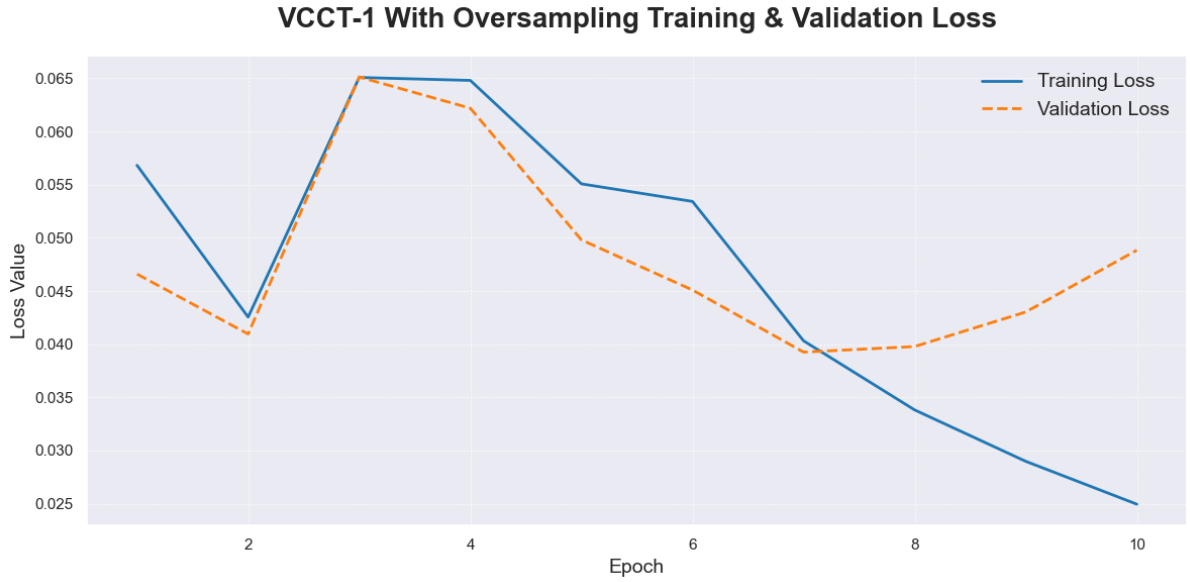

Figure 0.20 Loss plot for VCCT-1 with oversampling

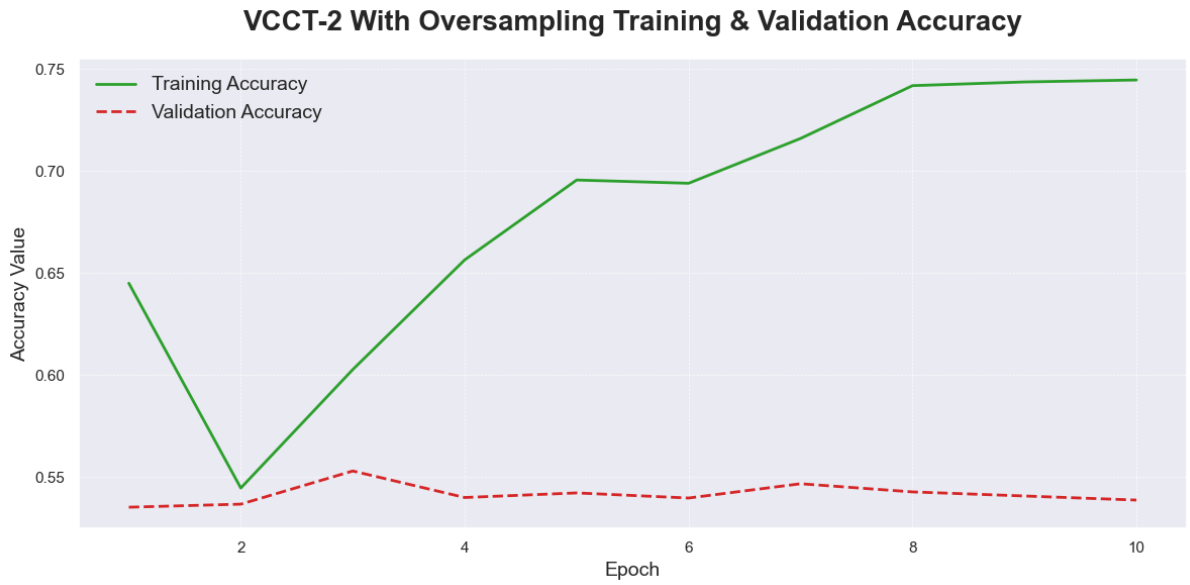

Figure 0.21 Accuracy plot for VCCT-2 with oversampling

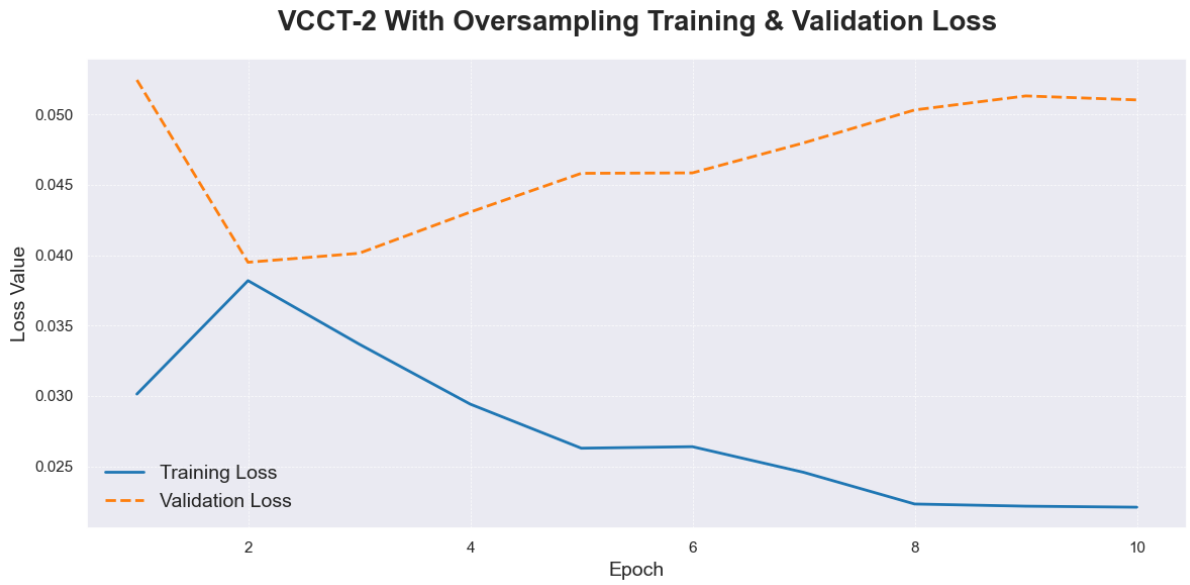

Figure 0.22 Loss plot for VCCT-2 with oversampling

Appendix B

Confusion Matrices

The vertical axis represents the true labels, while the horizontal axis denotes the predicted labels.

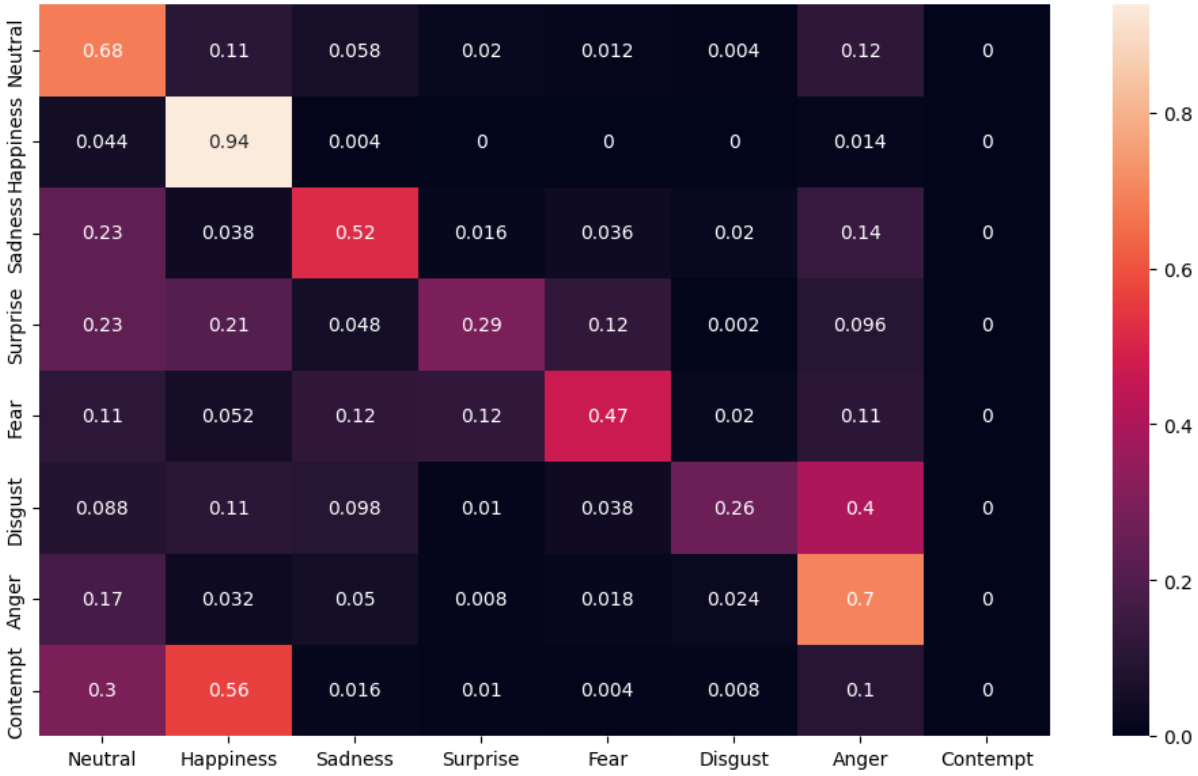

Figure 0.23 Confusion matrix of MCCT-1 without oversampling

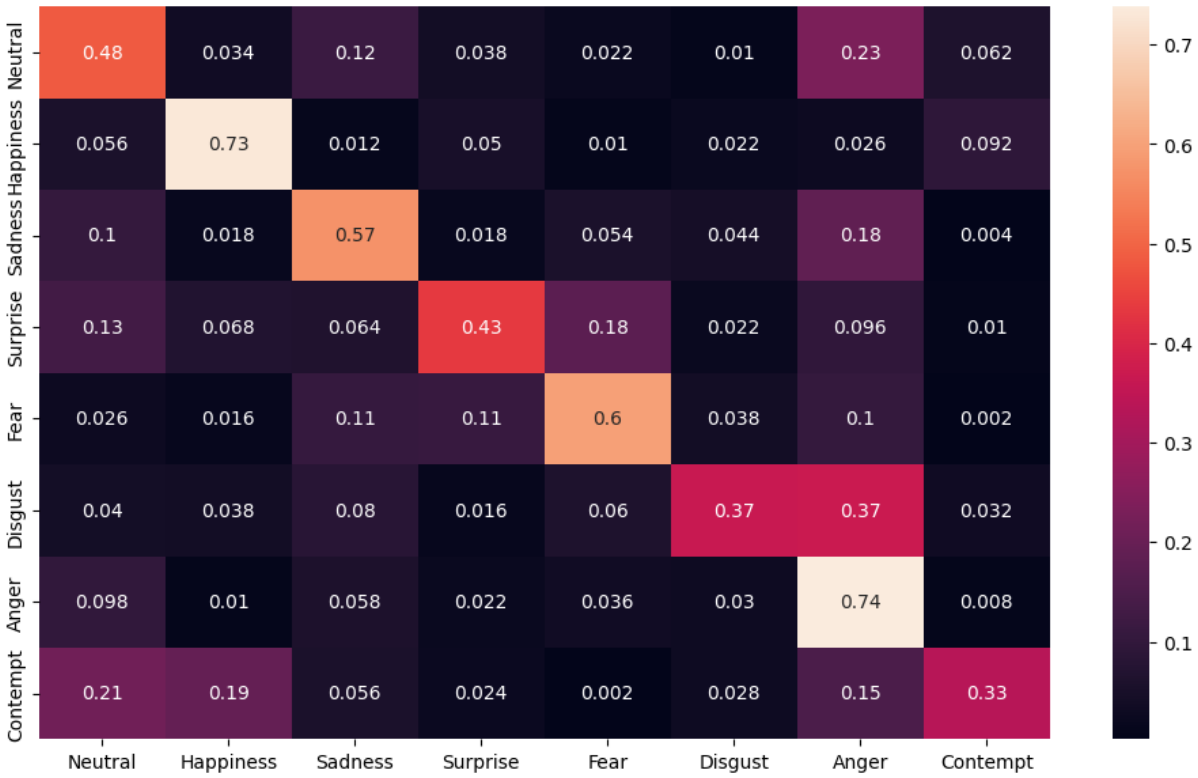

Figure 0.24 Confusion matrix of MCCT-1 with oversampling

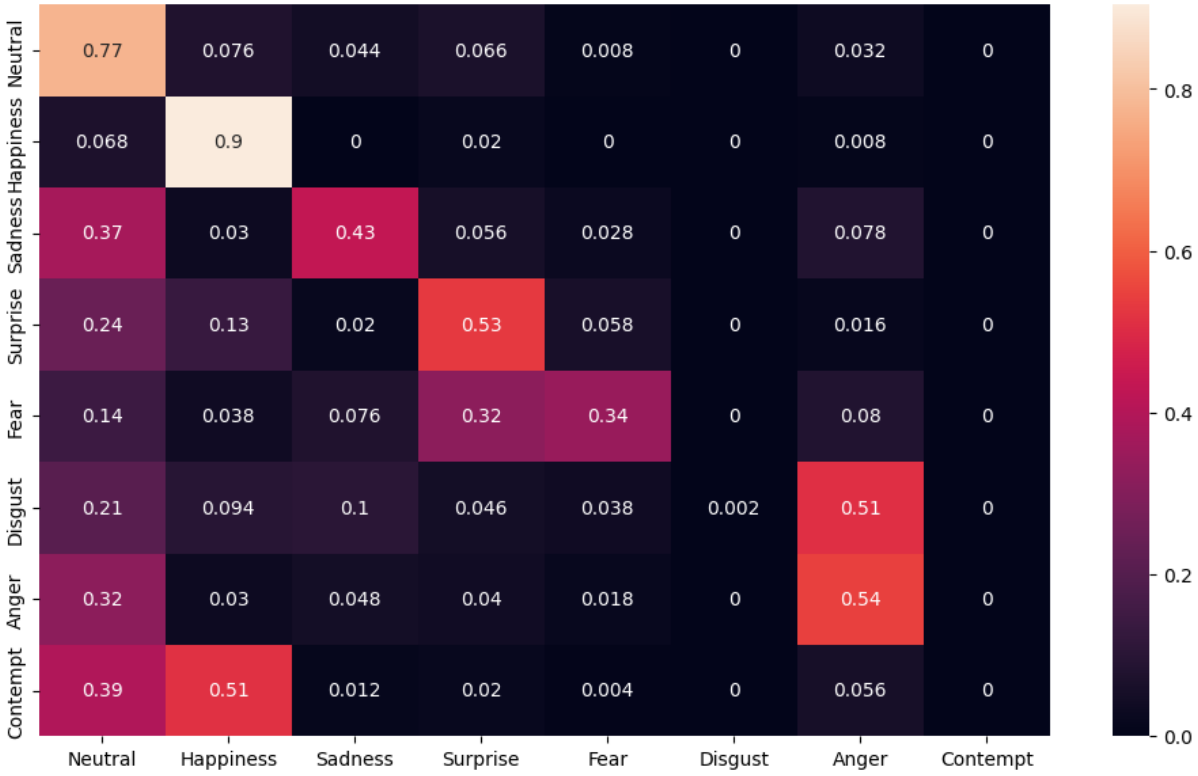

Figure 0.25 Confusion matrix of MCCT-2 without oversampling

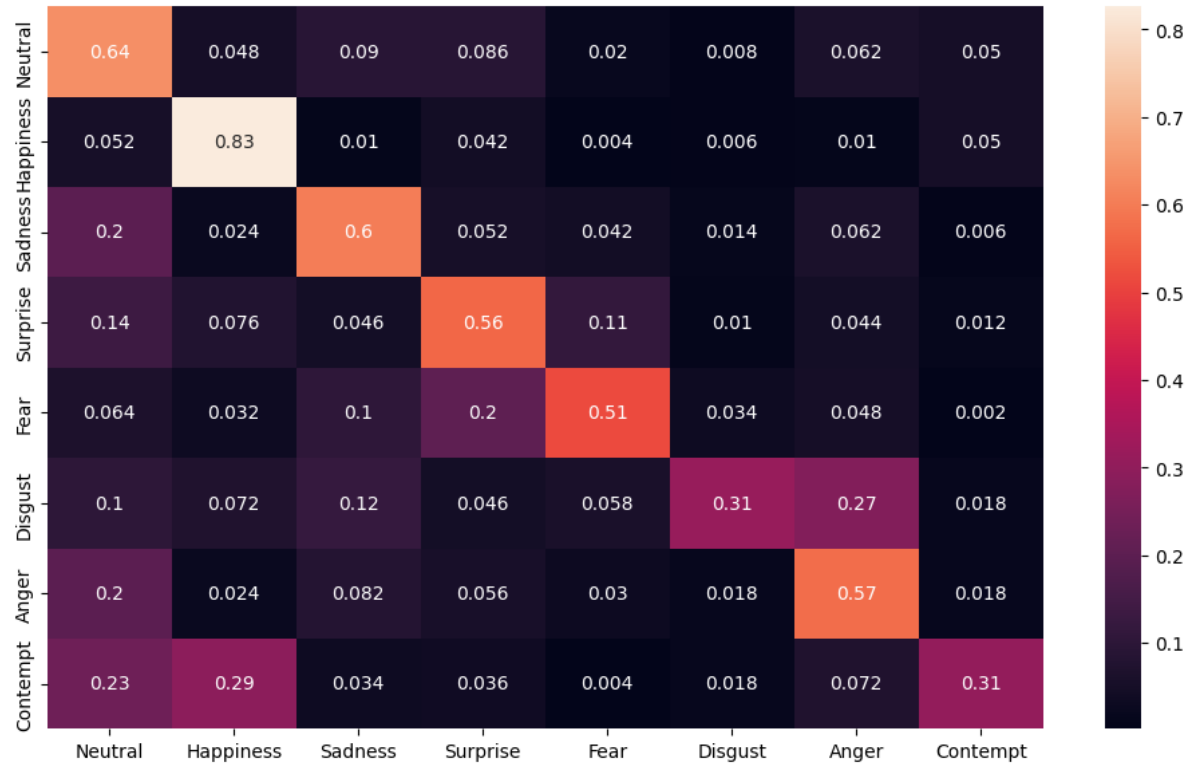

Figure 0.26 Confusion matrix of MCCT-2 with oversampling

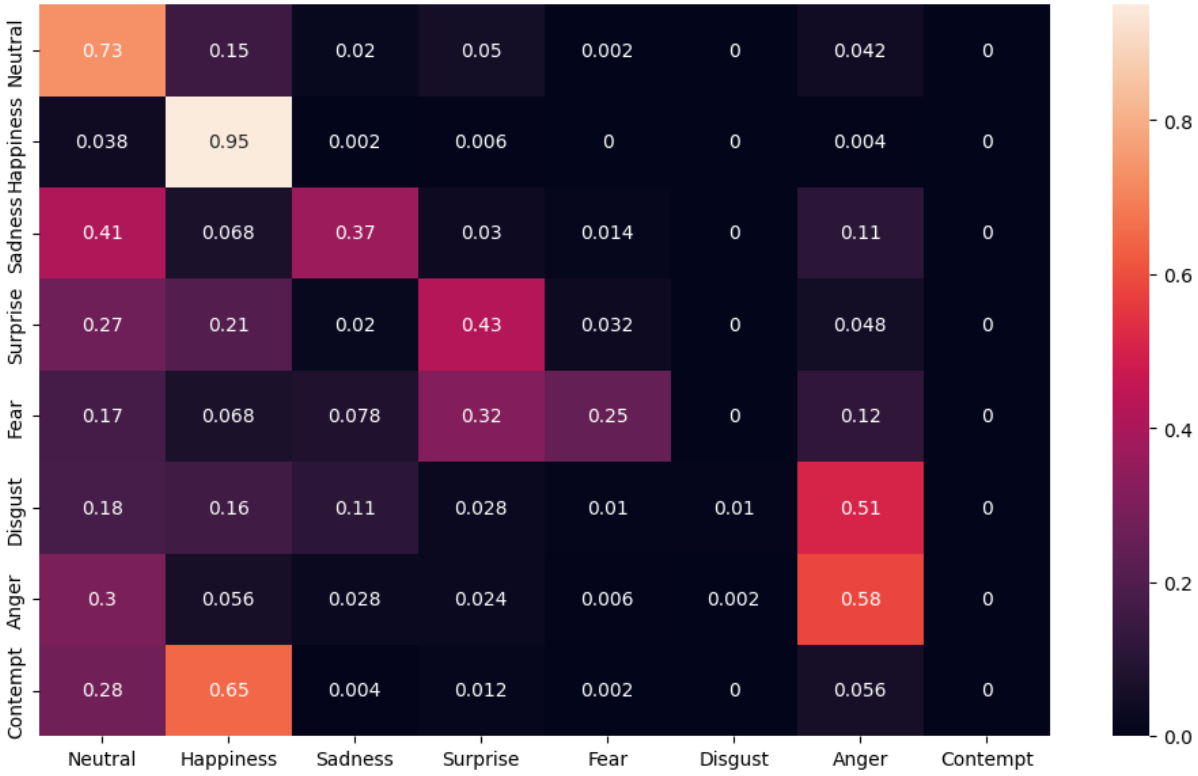

Figure 0.27 Confusion matrix of MCCT-3 without oversampling

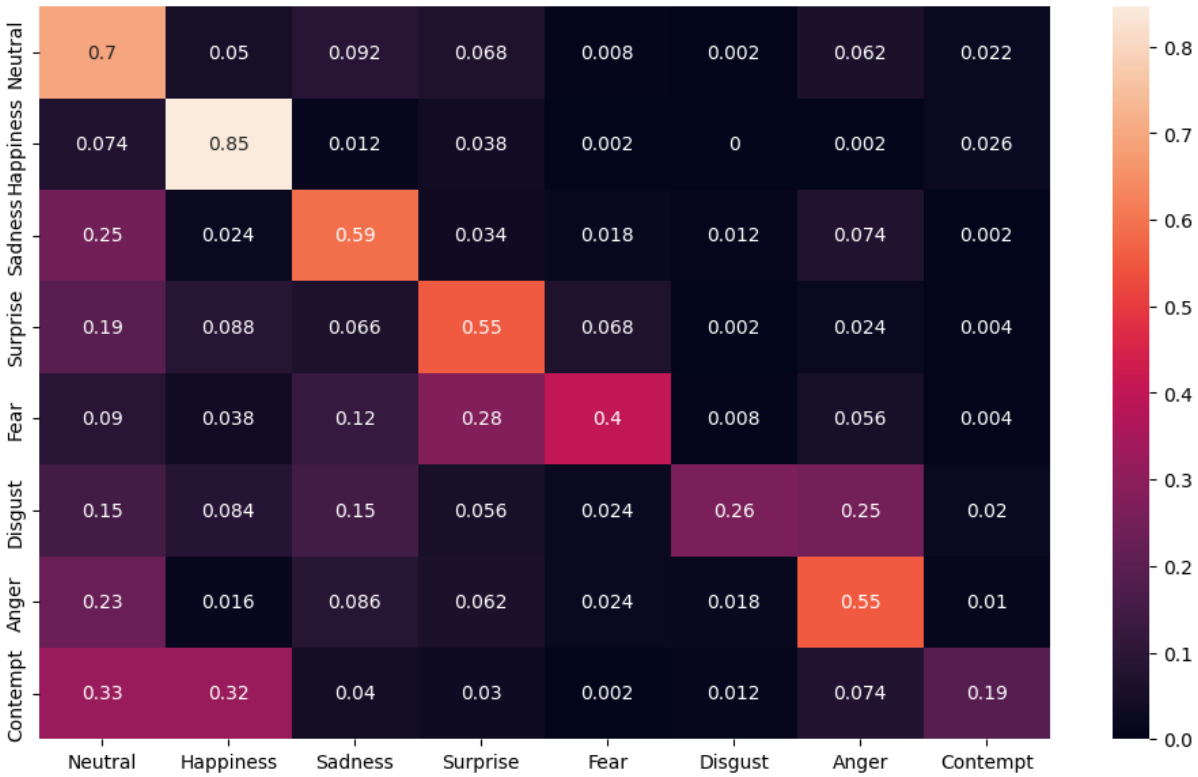

Figure 0.28 Confusion matrix of MCCT-3 with oversampling

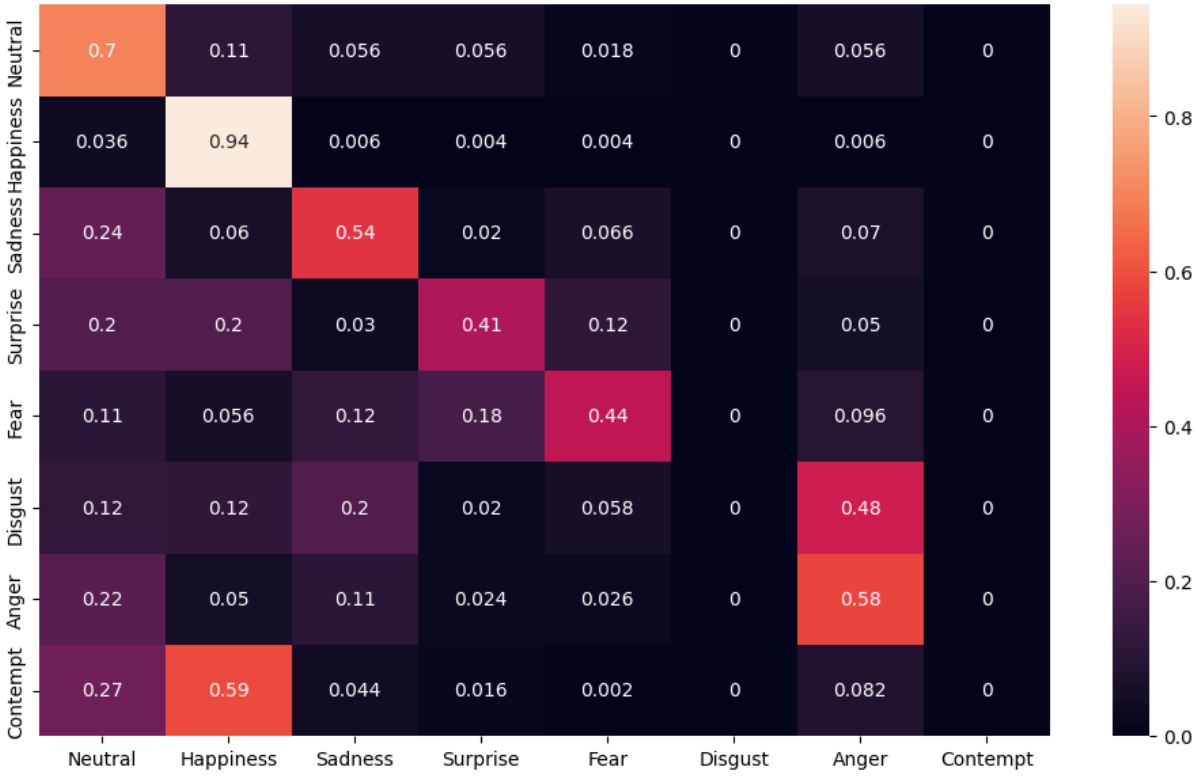

Figure 0.29 Confusion matrix of MCCT-6 without oversampling

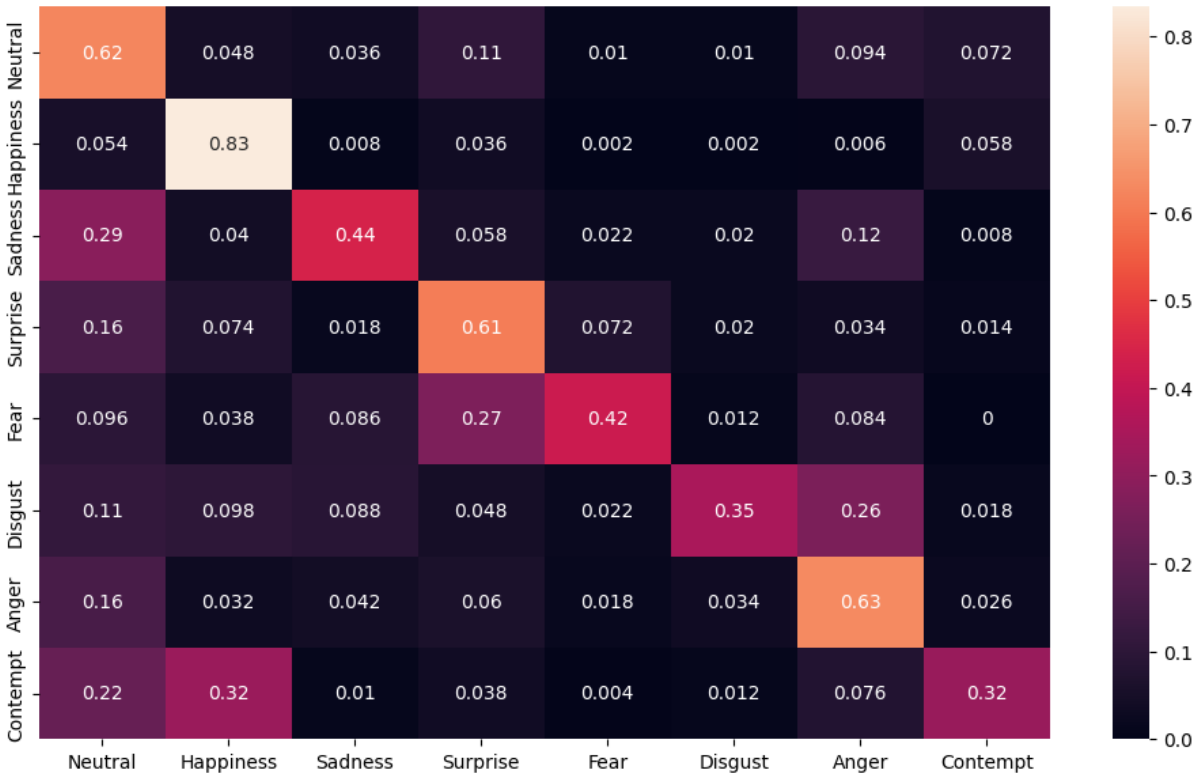

Figure 0.30 Confusion matrix of MCCT-6 with oversampling

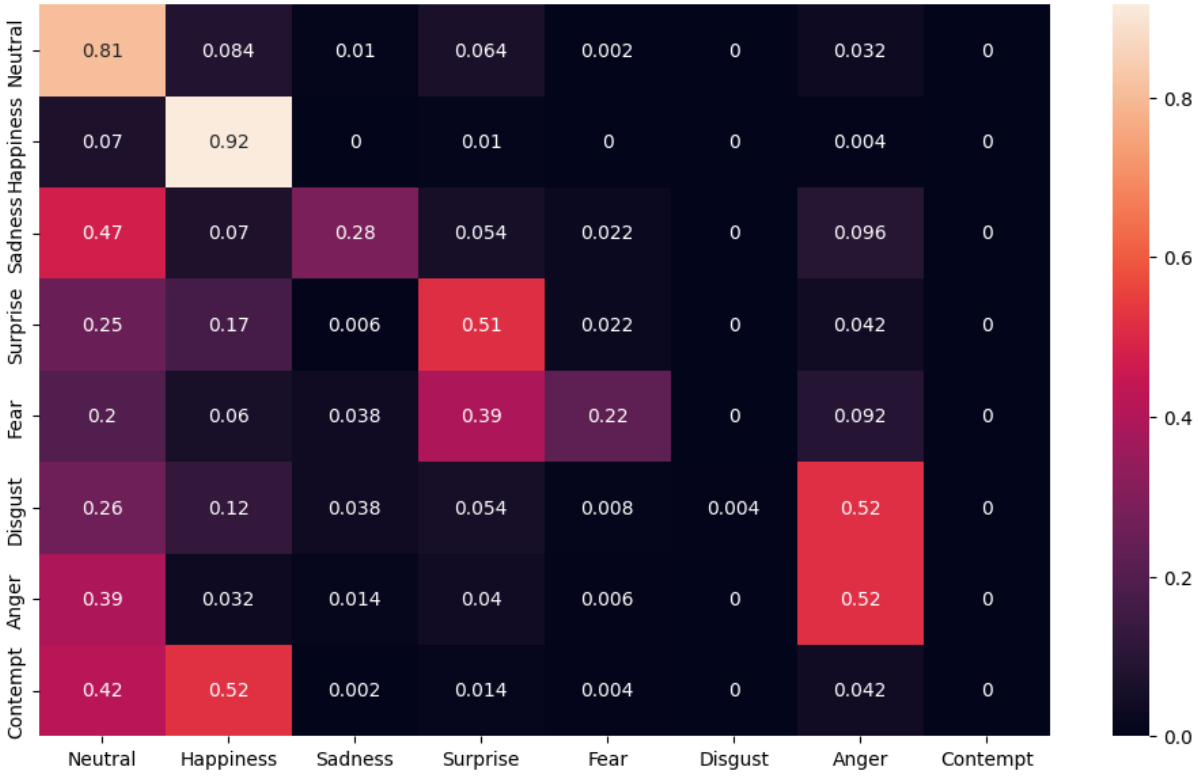

Figure 0.31 Confusion matrix of MCCT-7 without oversampling

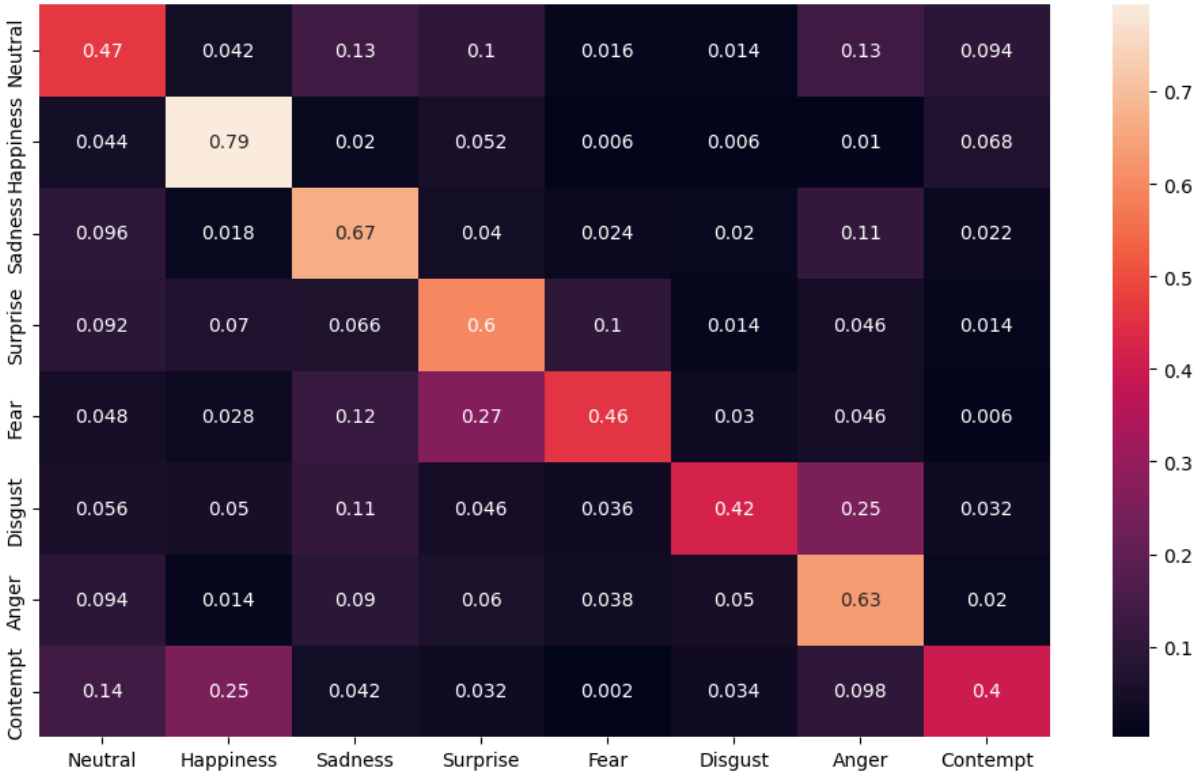

Figure 0.32 Confusion matrix of MCCT-7 with oversampling

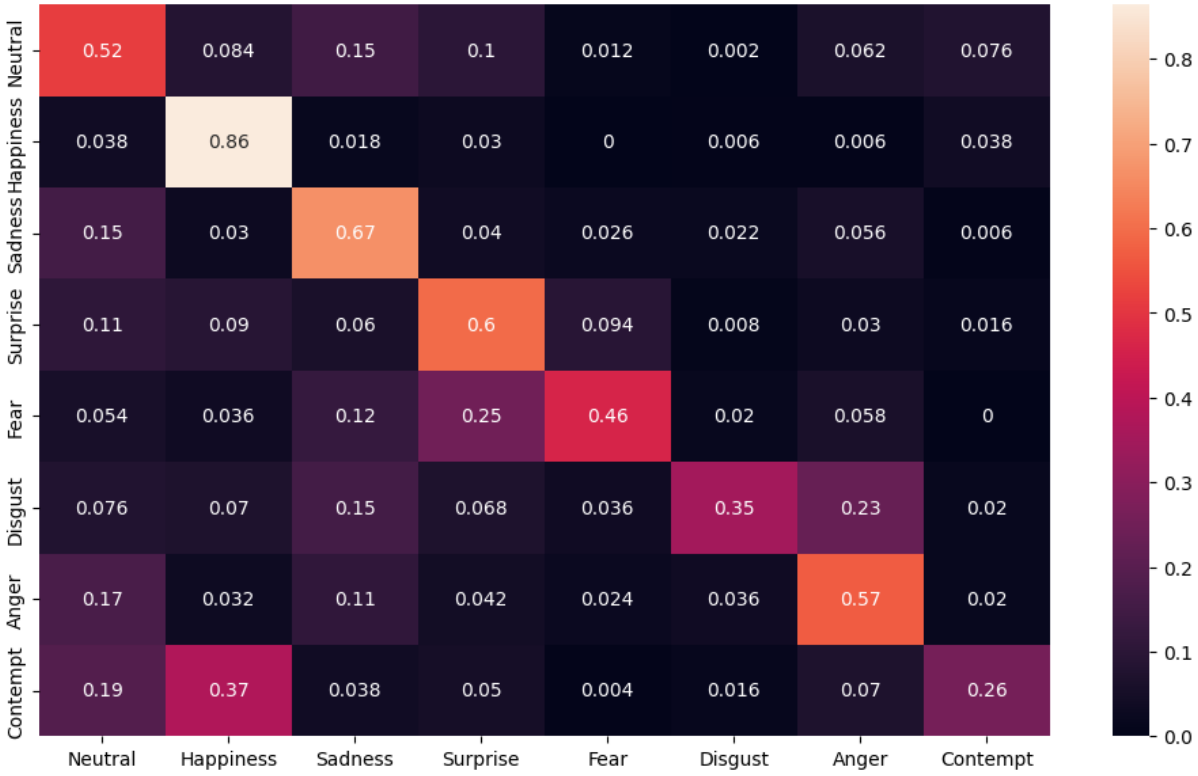

Figure 0.33 Confusion matrix of VCCT-1 with oversampling

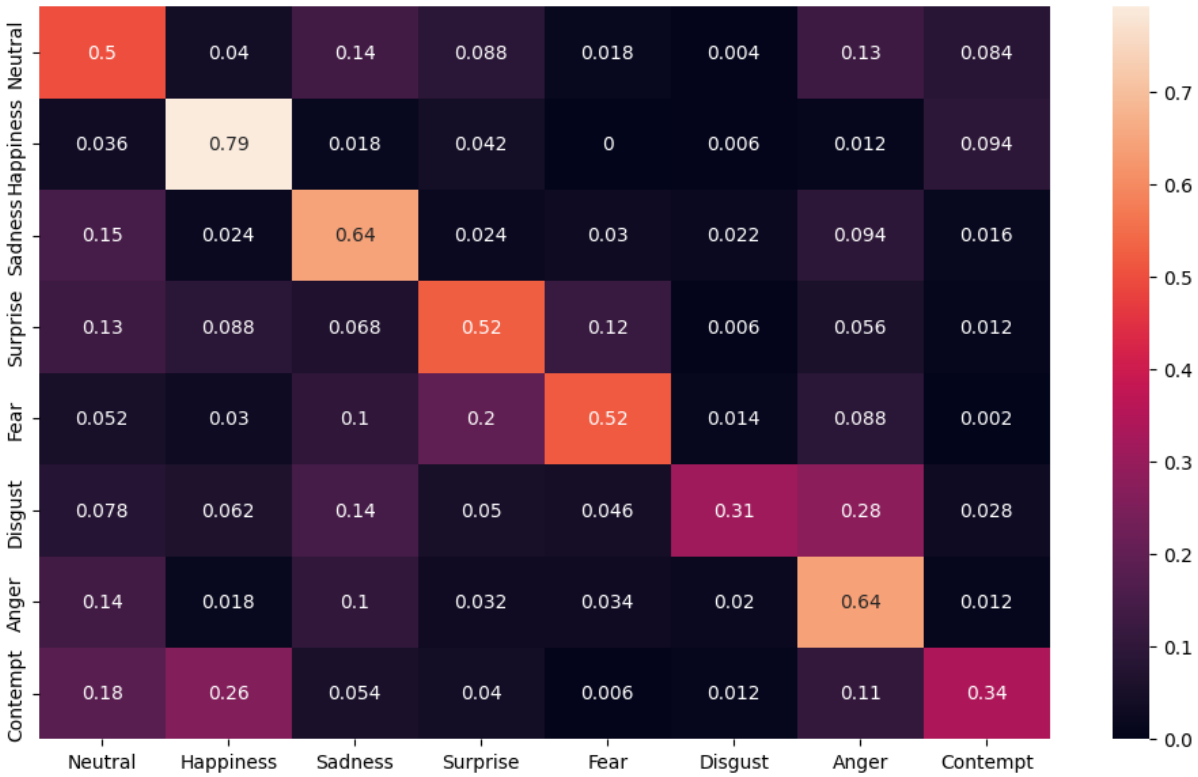

Figure 0.34 Confusion matrix of VCCT-2 with oversampling