Project Laboratory Report

Department of Telecommunications and Media Informatics

Author:	**Szenyán Zsombor**
Neptun:	`IOA33U`
Specialization:	**Infocommunication**
E-mail address:	**zsomborszenyan@edu.bme.hu**
Supervisor:	**Gyires-Tóth Bálint, PhD**
E-mail addres:	**tothb@tmit.bme.hu**
Co-Supervisor:	**Kalapos András**
E-mail addres:	**kalapos.andras@edu.bme.hu**

Title: Real-time Facial Attribute Recognition using Transformer networks

Task

In my project laboratory, I've researched the state-of-the-art solutions in the field of facial attribute recognition and improved on the efficiency of existing methods by leveraging Transformer networks. By harnessing the power of Transformer architecture, I've constructed a neural network that excels in real-time recognition of facial attributes. This new model not only ensures efficient and fast processing but also boasts accuracy levels on par with the current state-of-the-art systems. With its ability to swiftly and accurately recognize facial attributes, this project opens doors to real-time applications in various fields, from security systems to personalized user experiences.

2023/24 II. semester

1 Theory and previous works

1.1 Introduction

Facial attribute recognition, a fundamental task in computer vision, involves identifying and analyzing specific facial features such as gender, hair color, facial hair, and glasses. This capability has significant applications in security, surveillance, and personalized recommendations.

In security and surveillance, facial attribute recognition enhances monitoring systems and aids law enforcement by identifying individuals of interest and preventing security breaches. In personalized recommendations, it improves user experiences by tailoring suggestions based on attributes like age, gender, and emotional states, thus optimizing content delivery in e-commerce and entertainment.

This paper aims to provide a comprehensive overview of the current state of facial attribute recognition, reviewing methodologies, techniques, and challenges. Additionally, it explores leveraging transformer architectures—known for capturing long-range dependencies and modeling complex relationships—to enhance the performance, efficiency, and scalability of facial attribute recognition systems. The ultimate goal is to achieve real-time performance and unlock new applications.

1.2 Theoretical summary

Key concepts and methodologies in the field include:

- Facial Attribute Classification: This task involves categorizing facial images based on whether they possess specific predefined attributes or not. I will detail research on this topic in 1.3.1.
- Machine Learning Algorithms: Utilizing algorithms like Support Vector Machines (SVMs), decision trees, Convolutional Neural Networks (CNNs), and transformer architectures.
- Deep Learning: Compared to traditional machine learning methods, deep learning models automatically learn hierarchical features from raw data, eliminating the need for manual feature engineering.
- Convolutional Neural Networks O'Shea and Nash [2015] (CNNs): CNNs have revolutionized facial attribute recognition by enabling end-to-end learning from raw image data. They learn and extract features hierarchically, making them effective for attribute classification.
- Transformer Architectures Vaswani et al. [2017]: Initially developed for natural language processing tasks, transformer architectures have recently been applied to computer vision tasks, including facial attribute recognition. These architectures excel at capturing long-range dependencies and modeling complex relationships within the data, potentially leading to improved performance in facial attribute recognition tasks. The Vision transformer (ViT) Dosovitskiy et al. [2020] and the Swin Transformer Liu et al. [2021] are notable examples of transformer architectures that have shown promise in image recognition tasks.
- Multi-Task Learning: Training models to perform multiple tasks simultaneously, enhancing performance through shared representations.
- Data Augmentation: Applying transformations to training data to improve model robustness and generalization.

Overall, facial attribute recognition combines traditional computer vision techniques with advanced deep learning methodologies, increasingly utilizing CNNs and transformers to achieve state-of-the-art performance.

1.3 Starting point, previous works on this project

In recent years, transformer models have gained prominence in various natural language processing (NLP) tasks and have been increasingly applied to computer vision tasks, including facial attribute recognition. In this section, I review the related work on facial attribute recognition with a specific focus on the application of transformer models.

1.3.1 Earlier Approaches to Facial Attribute Recognition using Deep Learning

Deep learning has transformed facial attribute recognition, with Convolutional Neural Networks (CNNs) becoming key due to their ability to learn hierarchical features from raw data.

Hand and Chellappa [2016] have developed a multi-task deep convolutional neural network (MCNN) for attribute classification. The proposed architecture, along with an auxiliary network (AUX), significantly improves attribute classification accuracy compared to traditional methods. Their method achieved state-of-the-art performance on various attributes from the CelebA and LFWA datasets, with some attributes showing up to a 15% improvement over other methods. The MCNN architecture significantly reduces the number of parameters and training time required for attribute classification compared to independent CNNs, making it more efficient. The learned relationships among attributes in the auxiliary network provide insights into the correlations between different attributes, contributing to a better understanding of the underlying data.

Günther et al. [2016] have demonstrated that the application of data augmentation techniques, including random scaling, rotation, shifting, blurring, and horizontal flipping, not only does not compromise performance but also yields significant benefits. Their findings underscore the importance of leveraging data augmentation as a powerful strategy to enhance model robustness and performance in various tasks. By introducing variations in the training data through augmentation, models can learn more generalized features and exhibit improved performance across different scenarios.

Han et al. [2017] present a novel approach to heterogeneous face attribute estimation using Deep Multi-Task Learning (DMTL) with convolutional neural networks (CNNs). Unlike previous methods that either focused on estimating a single attribute or used separate models for each attribute without considering attribute correlation and heterogeneity, the proposed DMTL approach addresses these issues explicitly. The DMTL framework consists of shared feature learning for all attributes followed by category-specific feature learning for heterogeneous attribute categories. To handle attribute heterogeneity, the paper categorizes attributes into nominal vs. ordinal and holistic vs. local. Nominal attributes, such as race, are handled using classification schemes with cross-entropy loss, while ordinal attributes, such as age, are handled using regression schemes with Euclidean loss. Additionally, attributes are categorized as holistic or local based on whether they describe characteristics of the whole face or local facial components, respectively. The proposed DMTL approach outperforms state-of-the-art methods in face attribute estimation, as demonstrated through experiments on various benchmark datasets. The approach not only achieves high accuracy but also demonstrates excellent generalization ability, particularly in cross-database testing scenarios.

1.3.2 Transforming Image Recognition: A Comparative Review of Transformer Networks Versus CNNs

This section reviews recent efforts applying transformer models in computer vision, particularly facial attribute recognition, which has seen limited exploration.

One notable approach is the Vision Transformer (ViT) by Dosovitskiy et al. [2020], which treats image patches as tokens (similar to words in NLP) and processes them using a standard Transformer architecture. ViT achieves impressive results when pre-trained on large datasets and transferred to various image recognition benchmarks, surpassing state-of-the-art CNN-based models while requiring fewer computational resources. In contrast to CNNs, ViT exhibits less image-specific inductive bias, with only the MLP layers being local, while the self-attention layers are global. Positional information is preserved through the addition of learnable position embeddings.

The Swin Transformer by Liu et al. [2021] addresses the challenges of the ViTs by proposing a hierarchical Transformer architecture with shifted windows. This design enables modeling at various scales while maintaining linear computational complexity with respect to image size, while enhancing modeling power without sacrificing computational efficiency.

It has been demonstrated by Liu et al. [2022] that the transformer architecture can be effectively applied to facial attribute recognition tasks. Inspired by the visualization of feature attention map of different attributes, they naturally group attributes with similar attention regions into the same category. The proposed TransFa model utilizing the Swin Transformer architecture achieves state-of-the-art performance on the CelebA dataset, outperforming previous methods.

One drawback of the transformer architecture is its computational complexity, which is higher than that of CNNs. However, Li et al. [2022] have proposed the EfficientFormer, a lightweight transformer architecture that achieves competitive performance with state-of-the-art CNNs while being more efficient in terms of computational resources. With an inference time lower than the frametime of most displays or cameras, makes this model suitable for real-time applications. Li et al. [2023] later introduced the second version of the EfficientFormer, which further improves the performance of the model while maintaining its efficiency.

1.3.3 Details of the EfficientFormerV2

EfficientFormerV2 improves upon earlier transformer-based models like the ViT and the Swin Transformer, focusing on optimizing them for resource-constrained devices, especially mobile ones. Here, I'll detail the enhancements and strengths of EfficientFormerV2 compared to its predecessors.

The EfficientFormerV2 introduces several architectural improvements to achieve its goals. It employs a hierarchical design with four stages, capturing both local and global information efficiently.

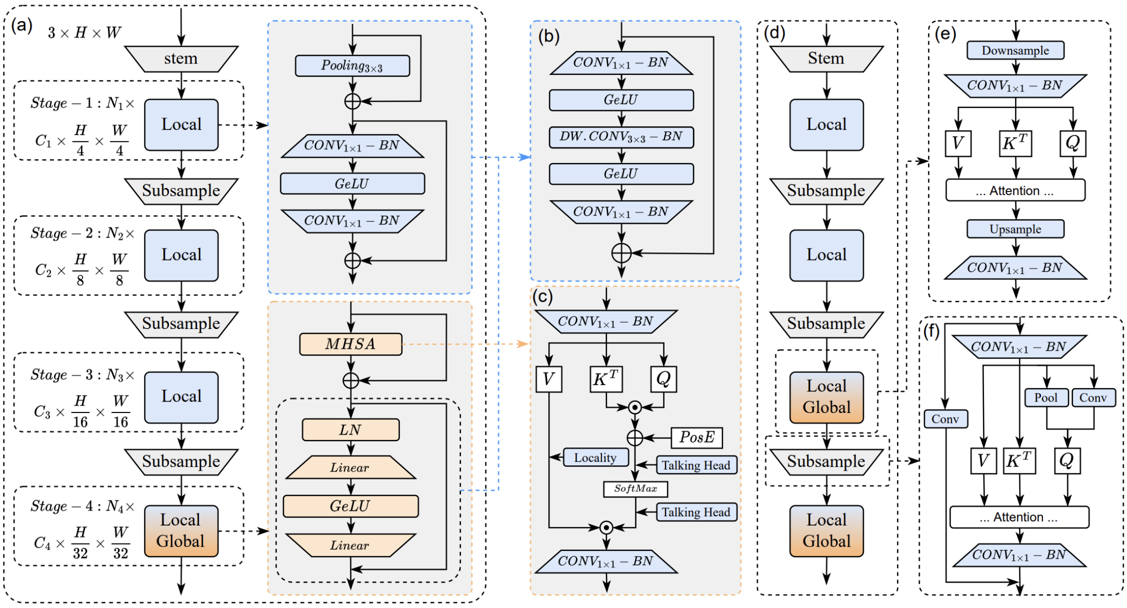


Figure 1: Overview of the EfficientFormerV2 improvements from Li et al. [2023]. (a) Network of EfficientFormer Li et al. [2022] that serves as a baseline model. (b) Unified FFN. (c) MHSA improvements. (d)&(e) Attention on higher resolution. (f) Dual-Path Attention downsampling.

The architecture starts with a convolutional stem for input embedding, ensuring efficient processing of input images. Instead of using separate local token mixers, EfficientFormerV2 incorporates depth-wise convolutions into the Feed Forward Network (FFN), reducing redundancy and improving performance with minimal parameter overhead. EfficientFormerV2 enhances the Multi-Head Self Attention (MHSA) mechanism by injecting local information into the Value matrix and enabling communication between attention heads. These modifications boost performance without significantly increasing model size or latency. To efficiently apply attention to higher-resolution features, EfficientFormerV2 introduces strategies like Stride Attention, which downsamples Queries, Keys, and Values to a fixed spatial resolution. This approach reduces latency while preserving competitive accuracy. EfficientFormerV2 introduces a novel downsampling strategy called dual-path attention downsampling. This method combines locality and global dependency, improving accuracy without compromising efficiency.

The search algorithm of EfficientFormerV2 considers both model size and inference speed as key factors, aiming to achieve Pareto optimality in terms of Mobile Efficiency Score (MES). This ensures that the resulting models are efficient for mobile deployment. With this search algorithm, the authors created 4 models with different trade-offs between model size and inference speed, allowing users to choose the best model for their specific requirements.

By combining architectural enhancements, optimization strategies, EfficientFormerV2 achieves higher performance than traditional lightweight CNNs while maintaining small model size and fast inference speed, making it suitable for real-time applications on mobile devices.

2 Own work on project

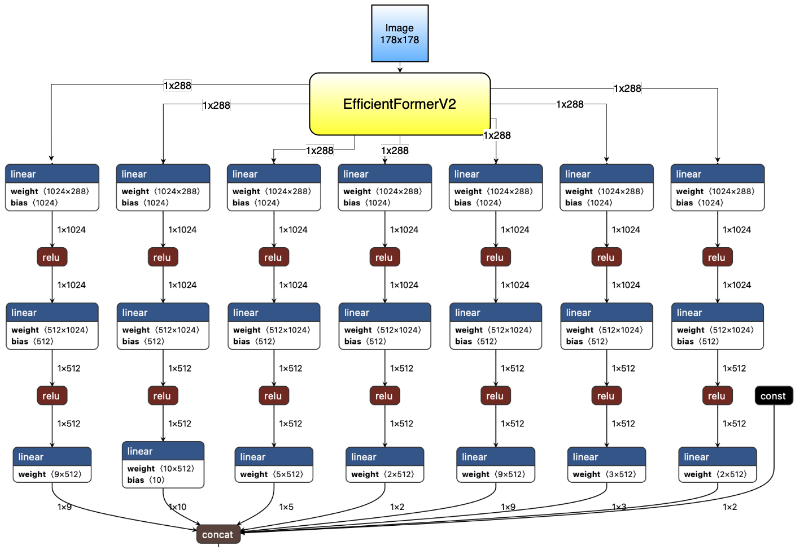


Figure 2: Model architecture overview.

2.1 Introduction to the dataset

In this section, I provide an overview of the dataset used to train my facial attribution recognition model. The CelebA dataset serves as the foundation for my model's training, validation, and testing phases. With a vast collection exceeding 200,000 images divided into training, validation, and test subsets, CelebA offers a robust platform for evaluating model performance and facilitating comparative analyses with other state-of-the-art approaches.

Each image within the CelebA dataset is standardized to a resolution of 218 by 178 pixels, ensuring consistency across the entire corpus. Notably, every image is labeled with 40 distinct facial attributes, providing rich annotations needed for supervised learning paradigms. It should be noted that the dataset's attributes are binary, with each attribute denoting the presence or absence of a specific facial feature or characteristic. Furthermore the attributes are not uniformly distributed, with some attributes being more prevalent than others. This imbalance poses a challenge for training models that must account for the varying frequencies of different attributes.

This comprehensive labeling scheme enables the development and refinement of highly accurate facial attribute recognition models, thereby enhancing the dataset's suitability for various machine learning applications.

2.2 Dataset preprocessing and augmentation

Preprocessing plays a crucial role in preparing the dataset for effective model training, while augmentation techniques further enhance the robustness and generalization capability of the trained model, essentially artificially inflating the size of the dataset. In this section, I detail the preprocessing and augmentation steps employed in my approach.

In regards to preprocessing, the following steps were undertaken:

Image Cropping to ensure compatibility with the transformer backbone, each image in the dataset underwent cropping. Initially sized at 218 by 178 pixels, the images were cropped to form perfect squares of 178 by 178 pixels. This transformation not only standardizes the input dimensions but also centers the facial features within the frame, facilitating more focused learning.

RGB to Float32 Conversion for quicker gradient descent. The color information encoded in the RGB (Red, Green, Blue) format using 8 bit floating point numbers was converted into floating-point values scaled between 0 and 1.

For augmentation I have utilized the PyTorch's transformation library's second version, providing a way to easily add common augmentation techniques.

I applied the ColorJitter transformation to introduce controlled variations in brightness, contrast, saturation, and hue. Specifically, I set the parameters to adjust brightness, contrast, and saturation by a maximum of 0.1, while allowing for a hue shift of up to 0.05. This augmentation strategy diversifies the dataset by simulating real-world variations in lighting conditions and color appearances, thereby enhancing the model's resilience to such variations during inference. Random Horizontal Flip to further enrich the dataset and promote model robustness, I incorporated random horizontal flipping. This simple yet effective transformation horizontally mirrors the images with a 50% probability during training. By introducing variations in facial orientations, this augmentation technique not only expands the dataset size but also encourages the model to learn invariant features, improving its ability to generalize across diverse facial poses and orientations.

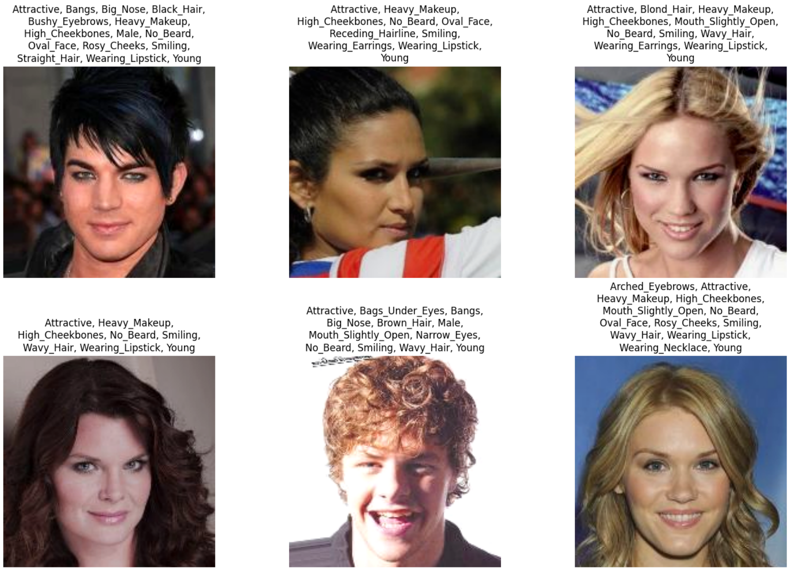

Attractive, Bangs, Big_Nose, Black_Hair, Bushy_Eyebrows, Heavy_Makeup, High_Cheekbones, Male, No_Beard, Oval_Face, Rosy_Cheeks, Smiling, Straight_Hair, Wearing_Lipstick, Young
Attractive, Heavy_Makeup, High_Cheekbones, No_Beard, Oval_Face, Receding_Hairline, Smiling, Wearing_Earrings, Wearing_Lipstick, Young
Attractive, Blond_Hair, Heavy_Makeup, High_Cheekbones, Mouth_Slightly_Open, No_Beard, Smiling, Wavy_Hair, Wearing_Earrings, Wearing_Lipstick, Young
Attractive, Heavy_Makeup, High_Cheekbones, No_Beard, Smiling, Wavy_Hair, Wearing_Lipstick, Young
Attractive, Bags_Under_Eyes, Bangs, Big_Nose, Brown_Hair, Male, Mouth_Slightly_Open, Narrow_Eyes, No_Beard, Smiling, Wavy_Hair, Young
Arched_Eyebrows, Attractive, Heavy_Makeup, High_Cheekbones, Mouth_Slightly_Open, No_Beard, Oval_Face, Rosy_Cheeks, Smiling, Wavy_Hair, Wearing_Lipstick, Wearing_Necklace, Young

Figure 3: Sample images from the CelebA dataset after preprocessing and augmentation.

By implementing these preprocessing and augmentation techniques, I optimized the dataset for training the model, fostering improved performance, generalization, and resilience to real-world variations.

2.3 Model architecture

The foundation of my facial attribution recognition model is built upon the EfficientFormerV2 backbone, because of its efficiency and effectiveness in processing visual data it makes it capable for efficient attribute recognition. Complementing this backbone architecture, my model employs multiple Feedforward Neural Network (FFN) heads to facilitate multi-task learning, inspired by Hand and Chellappa [2016], allowing for the simultaneous prediction of various facial attributes. This approach is enhanced by the hierarchical structure of the FFN heads, inspired by Han et al. [2017], which are strategically grouped based on distinctive facial regions, enabling targeted feature extraction and attribute prediction. These heads are strategically grouped based on distinctive facial regions, enabling targeted feature extraction and attribute prediction.

The EfficientFormerV2 backbone serves as the core of the model architecture, leveraging the advantages of transformer-based architectures for visual processing tasks. Renowned for its computational efficiency and scalability, EfficientFormerV2 efficiently captures intricate spatial dependencies within input images, facilitating superior feature representation and extraction.

My model incorporates seven FFN heads, each specializing in predicting attributes associated with specific facial regions (examples in parantheses):

- 9 Whole Face attributes (Attractive, Male, Smiling, Young)
- 10 Hair related attributes (Bald, Bangs, Blond Hair)
- 5 Eye region related attributes (Eyeglasses, Narrow Eyes, Arched Eyebrows)

- 2 Nose related attributes (Big Nose, Pointy Nose)
- 9 attributes in the Lips and Chin area (Big Lips, Mustache, No Beard)
- 3 attributes in the Cheeks and Ear area (High Cheekbones, Rosy Cheeks)
- 2 Neck region attributes (Wearing Necklace, Wearing Necktie)

Each FFN head comprises multiple layers designed to extract and process features relevant to the corresponding facial region. The architecture of each head follows a consistent pattern (visual representation in figure 2):

First Layer (1024 Neurons): The initial layer of each FFN head consists of 1024 neurons, serving as the primary feature extractor. This layer processes input features from the EfficientFormerV2 backbone, capturing region-specific information essential for attribute prediction.

ReLU Activation Layer: Following the first layer, a Rectified Linear Unit (ReLU) Agarap [2018] activation function is applied to introduce non-linearity into the model. The ReLU activation function is chosen for its simplicity and efficiency in promoting model convergence and performance.

Second Layer (512 Neurons): Subsequently, another fully connected layer comprising 512 neurons is employed to further refine the extracted features. This layer facilitates hierarchical feature abstraction, enabling the model to capture increasingly abstract representations of the input data. Another ReLU layer follows this layer.

Concatenation: The outputs of the second layer from each FFN head are concatenated into a unified feature vector, consolidating the extracted features from all facial regions.

Sigmoid Activation: Finally, a sigmoid activation function is applied to the concatenated feature vector to perform binary classification for attribute prediction. The sigmoid function produces probabilities indicating the likelihood of each attribute's presence, enabling the model to make informed predictions based on the learned features.

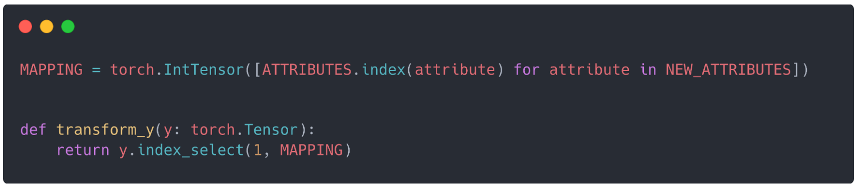
By adopting this multi-head FFN architecture, my model effectively leverages the hierarchical structure of facial attributes, enabling precise and comprehensive attribute prediction across diverse facial regions. This architecture promotes holistic understanding of facial characteristics while accommodating the unique attributes associated with each region, thereby enhancing the model's performance and versatility in facial attribution recognition tasks.

In total the model has 15,608,472 trainable with 12.6 million parameters in the EfficientFormerV2 backbone and the remaining parameters in the FFN heads.

2.4 Output transformation & hyperparameters

Upon generating predictions using the multi-head FFN architecture, the model's output must be transformed to align with the labels in the dataset. Given the introduction of regional grouping for attribute prediction, for further details please refer to section 2.3, the order of labels in the model's output may differ from that of the dataset labels. To facilitate accurate comparison, a transformation process is employed to reorder the model outputs accordingly. Please note that that the transformation process is not included within the model as it is unnecessary overhead during inference, but rather applied post-prediction, during training and evaluation, to ensure consistency with the dataset labels.

On the figure below the dataset's label order is shown as ATTRIBUTES and the model's output order is shown as NEW_ATTRIBUTES.



```
MAPPING = torch.IntTensor([ATTRIBUTES.index(attribute) for attribute in NEW_ATTRIBUTES])


def transform_y(y: torch.Tensor):
    return y.index_select(1, MAPPING)
```

Figure 4: Output transformation.

During experimentation, the impact of various hyperparameters on model performance, including the configuration of the EfficientFormer V2 backbone, have been tested.

Firstly I investigated the effect of switching between the various predefined configurations of the EfficientFormer V2 architecture. Despite initial expectations, transitioning from the S2 preset to a larger configuration (L2) did not yield improvements in model performance. Moreover, this modification significantly extended the training time and memory requirements, impeding overall training efficiency without commensurate gains in predictive accuracy.

Secondly the hyperparameters of the Two-Layer FFN have been investigated. Throughout experimentation, I explored variations in the width and depth of these FFN layers to ascertain their impact on model performance. Contrary to expectations, widening the FFN layers beyond the specified configuration did not yield appreciable enhancements in predictive accuracy. Additionally, attempts to improve model generalization through the incorporation of Dropout layers proved ineffective, as they only served to prolong convergence without yielding tangible improvements in performance.

Through experimentation and hyperparameter tuning, I optimized the model architecture and hyperparameters to strike a balance between predictive accuracy and training efficiency. The selected configuration reflects my efforts to maximize model performance while mitigating the risk of overfitting and training slowdowns.

2.5 Training

During the training phase, the model utilizes binary cross-entropy loss as the objective function to quantify the disparity between predicted attribute probabilities and ground truth labels. This loss function is particularly well-suited for binary classification tasks, such as facial attribute recognition, where each attribute is treated as a binary prediction task (presence or absence).

To optimize the model parameters and minimize the computed loss, the AdamW optimizer Loshchilov and Hutter [2017] is used. AdamW extends the Adam optimizer by incorporating weight decay regularization, thereby effectively preventing overfitting and enhancing model generalization. This optimizer's adaptive learning rate mechanism enables efficient convergence and robust performance across diverse datasets and architectures.

To strike a balance between training efficiency and memory utilization, a batch size of 64 is used. The value 64 is chosen as this is the highest batch size that can be accommodated within the available memory constraints, ensuring optimal hardware utilization during training.

I trained the model for 3 epochs on a single NVIDIA RTX 2070 SUPER GPU (8GB VRAM). The choice of 3 epochs is based on empirical observations during training, where the model's performance plateaued after the third epoch. Training takes approximately 30 minutes to complete, with each epoch lasting around 10 minutes.

2.6 Results

The proposed facial attribution recognition model demonstrates promising performance on the test dataset, achieving an accuracy of 90.9% with an average loss of 0.202585. This performance metric highlights the model's proficiency in correctly identifying facial attributes from input images.

Additionally, the model exhibits a recall rate of 73.0%, indicating its ability to effectively capture true positive instances among all actual positive cases.

Furthermore, the specificity of my model stands at 96.3%, underscoring its capability to correctly identify negative instances among all actual negative cases.

My model approaches state-of-the-art performance in facial attribute recognition, showcasing competitive accuracy while offering notable advantages in training speed, inference speed, and model size. Specifically, the model achieves an accuracy of 90.9%, only marginally lagging behind the performance of the model presented in the TransFA paper Liu et al. [2022], which reported an accuracy of 91.9%. However, my model outperforms TransFA in terms of training speed, inference speed, and model size as that uses the Swin Transformer architecture, which is inefficient compared to the EfficientFormerV2 architecture.

The DMM-CNN model by Mao et al. [2020] includes the parameter count and accuracy of their model, making it suitable for comparison. Their model used 360 million parameters compared to just 15.6 million in my model, while achieving an accuracy of 91.7%.

While there are no specific numbers reported for the specifics of the TransFA architecture regarding parameter count, training and inference speed, I suspect they are using the base Swin Transformer architecture as their paper reports using 12 Swin Transformer Layers, making just the backbone of their model 121 million parameters. Additionally their FFN heads use double the neurons in the first layer, making their head also use more parameters than mine.

On the radar plot below the performance of the model is visualized compared to the TransFA model. A Baseline performance is calculated using the distribution of attributes in the dataset, if the probability of an attribute occuring is less than 0.5 it is subtracted from 1, otherwise it is left as is. This subtraction is done to account for the imbalance in the dataset, as the model could achieve a high accuracy by just predicting the most common attributes.

The results of my study hold significant implications for the field of facial attribute recognition and related applications. Despite slightly trailing behind the state-of-the-art models in accuracy, my model offers substantial improvements in terms of efficiency and resource utilization. The enhanced training speed, inference speed, and reduced model size make my approach particularly appealing for real-world deployment in scenarios where computational resources are limited or efficiency is paramount.

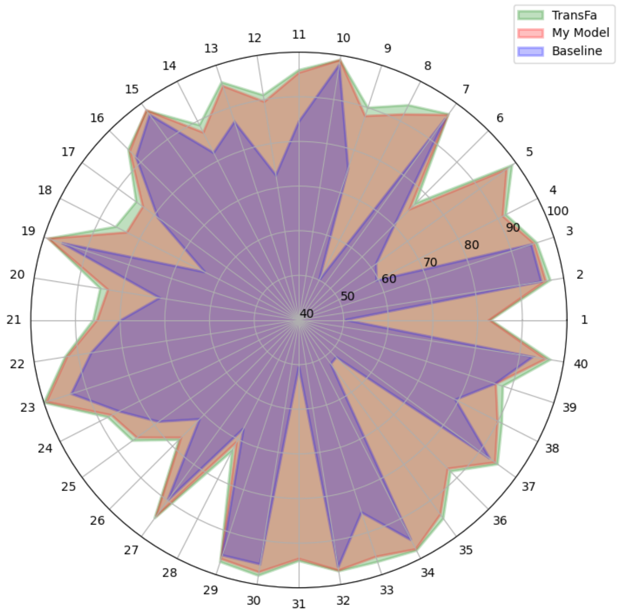


Figure 5: Radar plot per attribute performance visualization[1]

In conclusion, the proposed facial attribution recognition model demonstrates competitive performance in accurately identifying facial attributes from input images. While achieving a commendable accuracy of 90.9%, my model distinguishes itself through its efficiency gains in training and inference, as well as its compact model size. These findings underscore the potential of my approach to address the practical challenges associated with facial attribute recognition, paving the way for widespread adoption in various resource constrained applications.

2.7 iOS Demo Application

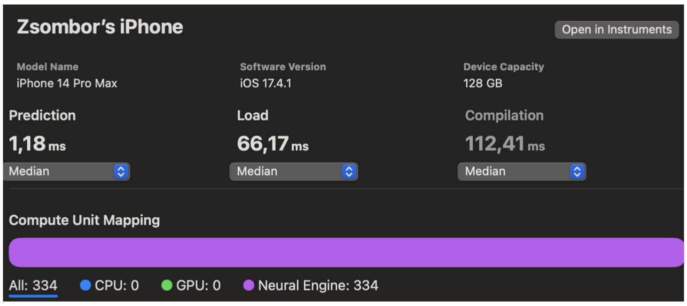
To demonstrate the real-time capability of my facial attribution recognition model, I developed an iOS demo application using SwiftUI. This application leverages the power of Core ML to deploy my PyTorch-based model directly onto iOS devices, enabling real-time inference on live camera feeds.

The first step in developing the iOS application involved converting the PyTorch model into a Core ML model using the CoreMLTools library. This conversion process facilitated seamless integration of my model into the iOS ecosystem, allowing for efficient execution on Apple's hardware platforms. CoreMLTools per-

[1] The attributes are as follows: Attractive, Blurry, Chubby, Heavy Makeup, Male, Oval Face, Pale Skin, Smiling, Young Bald, Bangs, Black Hair, Blond Hair, Brown Hair, Gray Hair, Receding Hairline, Straight Hair, Wavy Hair, Wearing Hat Arched Eyebrows, Bags Under Eyes, Bushy Eyebrows, Eyeglasses, Narrow Eyes Big Nose, Pointy Nose, 5 o'Clock Shadow, Big Lips, Double Chin, Goatee, Mouth Slightly Open, Mustache, No Beard, Sideburns, Wearing Lipstick High Cheekbones, Rosy Cheeks, Wearing Earrings Wearing Necklace, Wearing Necktie

forms optimizations tailored for inference purposes, ensuring optimal performance and resource utilization on iOS devices. CoreMLTools utilizes the following optimizations:

There are several optimizations that CoreMLTools performs during the conversion process, some of the most notable ones are: Quantization is used to convert the model to 16-bit floating point numbers, reducing the model size and improving inference speed. Layer fusion involves combining multiple operations or layers within the model into a single operation or layer, reducing the overall workload and improving efficiency. During conversion padding optimizations ensure efficient convolutions by optimizing padding operations. Dead code elimination removes unnecessary operations or layers from the model, reducing computational overhead and improving inference speed. Transpose optimization rearranges data in the model to improve memory access patterns and reduce latency.



Zsombor's iPhone

Model Name: iPhone 14 Pro Max

Software Version: iOS 17.4.1

Device Capacity: 128 GB

Prediction: 1,18 ms (Median)

Load: 66,17 ms (Median)

Compilation: 112,41 ms (Median)

Compute Unit Mapping

All: 334 ● CPU: 0 ● GPU: 0 ● Neural Engine: 334

Figure 6: iOS inference benchmark.

The converted Core ML model exhibited remarkable efficiency, achieving inference speeds as low as 1.18 milliseconds on the iPhone 14 Pro Max. After the conversion process I reran the model on the CelebA dataset to ensure that the conversion process did not affect the model's performance. The model miraculously slightly improved its accuracy, achieving a score of 91% on the test dataset, demonstrating the robustness and reliability of the Core ML conversion process. The accuracy improvement can be attributed to the quantization process which can sometimes improve the model's performance by rounding off the weights and biases to the nearest 16-bit floating point number in a way that improves the model's performance. Crucially, the model leveraged the device's neural engine for each calculation, harnessing specialized hardware acceleration to expedite inference tasks while conserving battery life.

The iOS demo application features a user-friendly interface designed using SwiftUI, Apple's declarative framework for building user interfaces. Upon launching the application, users are presented with a simple yet engaging interface, showcasing a circular frame in the center of the screen. As users position their head within the designated circular frame, the application initiates real-time attribute recognition by capturing frames from the device's camera and passing them through the Core ML model. The model analyzes each frame and identifies facial attributes deemed to be present, providing instant feedback on the bottom of the screen. By double tapping on the screen, users may stop the camera feed and model inference, allowing them to view the detected attributes in detail. Double tapping again resumes the camera feed and model inference.

Thanks to the low inference speed and efficient hardware utilization of the Core ML model, the iOS demo application delivers seamless performance with no dropped frames. Users can experience smooth and uninterrupted attribute recognition.

My iOS demo application showcases the practical implications of a real-time facial attribution recognition model, extending its capabilities to mobile platforms and empowering users with real-time attribute recognition on their iOS devices.

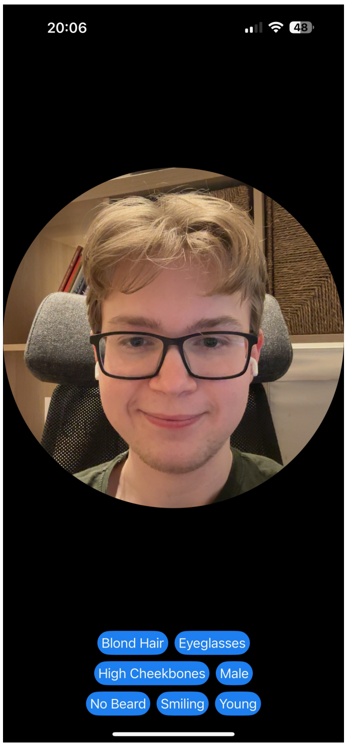


Figure 7: iOS demo application interface.

2.8 Summary

During my project laboratory I researched state-of-the-art approaches in the field of facial attribute recognition, identified key challenges, and proposed novel solutions to enhance model efficiency while maintaining performance. The study begins by reviewing earlier approaches to facial attribute recognition using deep learning, highlighting the contributions of convolutional neural networks (CNNs) and multi-task learning frameworks.

Moving forward, the paper discusses recent advancements in transformer-based architectures, such as the Vision Transformer (ViT) and Swin Transformer, in computer vision tasks and their limited application to facial attribute recognition. It then delves into the development of TransFa, a transformer-based model utilizing the Swin Transformer architecture, which achieves state-of-the-art performance on the CelebA dataset.

The study addresses the computational complexity challenge inherent in transformer architectures by introducing the EfficientFormerV2, a lightweight transformer model optimized for efficient deployment on resource-constrained devices, particularly mobile devices. In my approach I leverage the EfficientFormerV2 backbone to develop a facial attribute recognition model that excels in real-time performance, efficiency, and accuracy.

Furthermore, the paper provides a detailed overview of my use of data augmentation for training, enhancing dataset robustness and model performance. It then describes the model architecture, which incorporates the EfficientFormerV2 backbone and multiple Feedforward Neural Network (FFN) heads for multi-task learning.

The study also discusses output transformation and hyperparameters optimization, outlining the choice of objective function, optimizer, batch size, and training epochs. The model's performance is evaluated against state-of-the-art approaches in facial attribute recognition, highlighting its competitive accuracy, while signifanctly increasing training speed, inference speed, and decreasing model size.

In conclusion, the paper presents an intuitive iOS demo application showcasing real-time attribute recognition using the developed facial attribute recognition model. Leveraging Core ML and SwiftUI, the application delivers seamless performance on iOS devices, demonstrating the practical implications of transformer-based models in real-world scenarios.

References

Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *CoRR*, abs/1511.08458, 2015. URL http://arxiv.org/abs/1511.08458.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*, abs/2103.14030, 2021. URL https://arxiv.org/abs/2103.14030.

Emily M. Hand and Rama Chellappa. Attributes for improved attributes: A multi-task network for attribute classification. *CoRR*, abs/1604.07360, 2016. URL http://arxiv.org/abs/1604.07360.

Manuel Günther, Andras Rozsa, and Terrance E. Boult. AFFACT - alignment free facial attribute classification technique. *CoRR*, abs/1611.06158, 2016. URL http://arxiv.org/abs/1611.06158.

Hu Han, Anil K. Jain, Shiguang Shan, and Xilin Chen. Heterogeneous face attribute estimation: A deep multi-task learning approach. *CoRR*, abs/1706.00906, 2017. URL http://arxiv.org/abs/1706.00906.

Decheng Liu, Weijie He, Chunlei Peng, Nannan Wang, Jie Li, and Xinbo Gao. Transfa: Transformer-based representation for face attribute evaluation, 2022.

Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. Efficientformer: Vision transformers at mobilenet speed. *Advances in Neural Information Processing Systems*, 35:12934–12949, 2022.

Yanyu Li, Ju Hu, Yang Wen, Georgios Evangelidis, Kamyar Salahi, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Rethinking vision transformers for mobilenet size and speed. In *Proceedings of the IEEE international conference on computer vision*, 2023.

Abien Fred Agarap. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018. URL http://arxiv.org/abs/1803.08375.

Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. URL http://arxiv.org/abs/1711.05101.

Longbiao Mao, Yan Yan, Jing-Hao Xue, and Hanzi Wang. Deep multi-task multi-label CNN for effective facial attribute classification. *CoRR*, abs/2002.03683, 2020. URL https://arxiv.org/abs/2002.03683.