OULUN AMMATTIKORKEAKOULU

Dang Chau Nguyen

AGE ESTIMATION FROM FACIAL IMAGES USING MACHINE LEARNING

AGE ESTIMATION FROM FACIAL IMAGES USING MACHINE LEARNING

Dang Chau Nguyen
Bachelor's Thesis
Spring 2023
Information Technology
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Bachelor of Engineering, Information Technology

---

Author(s): Dang Chau Nguyen
Title of the thesis: Age estimation from facial images using Machine Learning
Supervisor: Lasse Haverinen
Term and year of thesis completion: Spring 2023 Pages: 49 + 1 appendix

---

In today's era of online services, there is a growing need to verify user age to comply with legal requirements. However, traditional methods requiring collecting personal identification data may compromise user privacy and security. In response to the challenge, this thesis presents an approach based on machine learning for age estimation, designed to predict user age from facial images without collecting all sensitive user data. As a result, this proposed approach reduces the resources required to process and store sensitive user information, reducing the potential impact of data breaches.

The proposed approach uses Machine Learning models, such as ResNet and MobileNet, to process facial images and predict user age. The approach is evaluated using various performance metrics. Despite initial promising results, the proposed approach did not reach the desired accuracy due to overfitting issues, signaling the need for further refinement and research. Nonetheless, the approach promises reduced resources needed to process private information, potentially lowering significant costs, and offering efficiency benefits for online services.

In conclusion, while the proposed machine learning based age verification approach showed potential, it did not fully achieve the expected outcomes in this study, underlining the need for continued investigation. Future research should also consider potential applications in other forms of biometric identification while prioritizing user privacy and security.

Keywords: age estimation, facial images, computer vision, deep learning, resnet-50, mobilenet

ACKNOWLEDGEMENTS

I am grateful to Oulu University of Applied Sciences - OAMK, where this thesis was conceived and nurtured to completion. The institutional support and resources were invaluable in facilitating the thesis work.

I am indebted to my supervisor, Mr. Lasse Haverinen, for his relentless guidance and invaluable suggestions that have significantly improved this thesis's quality. His coaching on best practices for thesis writing has been a cornerstone of this work.

I am deeply grateful to my best friend, Hoang Ngan Nguyen, a brilliant Ph.D. student at King Abdullah University of Science and Technology. Our friendship and our professional paths mirrored in the form of a pair of entangled quantum particles. As she excels in academia, I thrive in the IT industry, and our successes are interlinked and inspiring. Ngan's insightful revelation about the progress of Machine Learning, especially Deep Learning, coupled with her explorations into the scientific research related to understanding artificial neural network layers, sparked a renewed passion within me. This inspiration attracted me back into the fascinating world of Deep Learning.

Special appreciation to my friends who have supported me on this journey. Thuong Khanh Tran, a doctoral researcher at the Center for Machine Vision and Signal Analysis, University of Oulu, has been a source of inspiration with his influential work in Computer Vision and AI. Trideptrai Hong Tri Nguyen, a doctoral researcher at the Center for Ubiquitous Computing, University of Oulu, provided unwavering support and displayed a dedication to quality akin to a SEAL officer, present when needed and consistently delivering under pressure.

Lastly, my deepest gratitude goes to my family. To my mother and my wife, Hoang Anh Nguyen, your unwavering faith, constant love, and support have been the bedrock upon which this journey was built. Your encouragement and belief in me have been my greatest sources of strength.

CONTENTS

ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
AUC-ROC	Area under the receiver operating characteristic curve
AWS	Amazon Web Service
CNN	Convolutional Neural Network
CPU	Central processing units
GCP	Google Cloud Platform
GPT	Generative Pre-trained Transformer
GPU	Graphics processing units
ML	Machine learning
MAE	Mean absolute error
MSE	Mean squared error
PiT	Pooling-based Vision Transformer
RMSE	Root mean squared error
TPU	Tensor processing unit
UTK	University of Tennessee, Knoxville
VGG	Visual Geometry Group
ViT	Vision Transformers
VM	Virtual Machine
VMAGE	VGG-Face2 Mivia Age Estimation

VOCABULARY

Backend	The hidden part of a software application that manages processing and business logic
eCommerce	The buying and selling of goods and services through electronic platforms such as websites
Epoch	A single pass through the entire training dataset during model training
Frontend	The user facing part of a software application that handles the presentation and interaction with the user
Instance	A specific computational resource used for running tasks or applications. Typically refer to a virtual machine in cloud computing.

1 INTRODUCTION

Age verification is critical for many online services to comply with legal and ethical standards. In particular, many industries, such as gambling, tobacco, and alcohol, are heavily regulated and require businesses to verify the age of their customers in order to comply with laws and regulations. Age verification helps businesses comply with these regulations and avoid legal penalties.

Machine learning is a powerful field that can potentially transform various industries and applications. Predictive analytics using machine learning can be used to analyze large amounts of data and predict future events. This can be applied in many industries, such as finance, healthcare, and marketing, to help businesses make more informed decisions. In addition, machine learning is used in natural language processing to understand and generate human language, which has many applications such as chatbots, virtual assistants, and language translation. Moreover, machine learning can analyze and interpret visual data, such as images and videos, in applications such as facial recognition, object detection, and autonomous vehicles.

This thesis proposes an approach for age verification that utilizes a combination of deep learning models, such as ResNet and MobileNet. By processing facial images, these models can accurately predict user age while minimizing the need for collecting sensitive personal identification data. The proposed approach is evaluated using a variety of performance metrics. While initial results demonstrated potential, the approach fell short of the desired accuracy and stability due to overfitting issues. This underlines the necessity for further research and refinement. Despite the challenges, the strategy's potential lies in its capacity to reduce the resources required for processing sensitive personal data, potentially leading to significant cost and efficiency benefits for online services.

The structure of the thesis: Section 2 is about theory and foundation of machine learning. The detailed analysis of this work is in section 3, while section 4 indicates implementation. Section 5 concludes this work and discusses future works.

2 THEORY AND FOUNDATION

2.1 Machine Learning

Machine learning (ML) is a subset of artificial intelligence that focuses on developing algorithms and statistical models. In machine learning, a model is trained on a dataset, and then uses this model to make predictions about new data. [1].

ML idea was developed early in the 1950s, with engineers making ideas about self-teaching computers during this time period. One of the most influential and widely cited definitions of machine learning was given by Tom M. Mitchell, who stated that: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." [2, p. 2]. This definition captures the essence of what machine learning algorithms do: learn from data to perform specific tasks better. ML continues evolving through many steps, from finding patterns with rudimentary reinforcement learning to evolving with neural networks in the 1970s [3]. The advent of cloud computing in the 2010s made it easier to access powerful computing resources without huge investments, enabling faster and more scalable machine learning models. Other significant factors were the development of newer ML algorithms that made use of GPU and customized hardware such as TPU, which boosted the performance and efficiency of training deep learning models for complex tasks. During the early 2020s, AutoML emerged as a dominant trend [4], allowing data scientists, analysts, and developers to build high-quality machine learning models with less effort and more efficiency by automating the selection of algorithms and parameters. A significant breakthrough in 2023 is the development of ChatGPT unlocking emergent abilities [5] with remarkable sparks of Artificial General Intelligence [6], the ultimate goal of machine learning.

Machine learning is being used in a wide range of applications, from image and speech recognition to recommendation systems, fraud detection, and autonomous vehicles. As data and computing power continue to grow, the potential applications for machine learning are expanding rapidly, making it an exciting and rapidly evolving field.

2.1.1 Machine Learning challenges

In machine learning, generalization refers to the model's ability to perform well on new, unseen inputs, beyond just the training set. The measure of this ability is the generalization or test error. Machine learning aims to minimize this error, distinguishing it from simple optimization tasks.

Two critical issues in machine learning are underfitting and overfitting [7]. Underfitting occurs when the model fails to minimize the error value on the training set, while overfitting is when the difference between the training error and test error is substantial. A model's capacity is its potential to adapt to a range of functions, impacts the likelihood of underfitting or overfitting. Models of low capacity may underfit, struggling with the training set. In contrast, those with high capacity are subjected to overfit by excessively memorizing from the training set to the degree that impairs their performance on the test set.

2.1.2 Machine Learning models

Machine learning models are mathematical functions that learn from data and make predictions or decisions [8]. There are different ways to represent these functions, depending on the type and complexity of the problem. Some common forms of machine learning models are:

- linear regression model (see Figure 1) finds only one line that best fits the given dataset using a mathematical function.
- decision trees (see Figure 2) use a hierarchical structure of rules to split the data into smaller subsets.
- support vector machines find a hyperplane that separates the data into different classes
- random forests combine multiple decision trees and average their outputs.
- k-nearest neighbors classifier (see Figure 3) assign a label to a new data point based on the labels of its closest neighbors.
- artificial neural networks use multiple layers of non-linear transformations to extract high-level features from the data.

and so on.

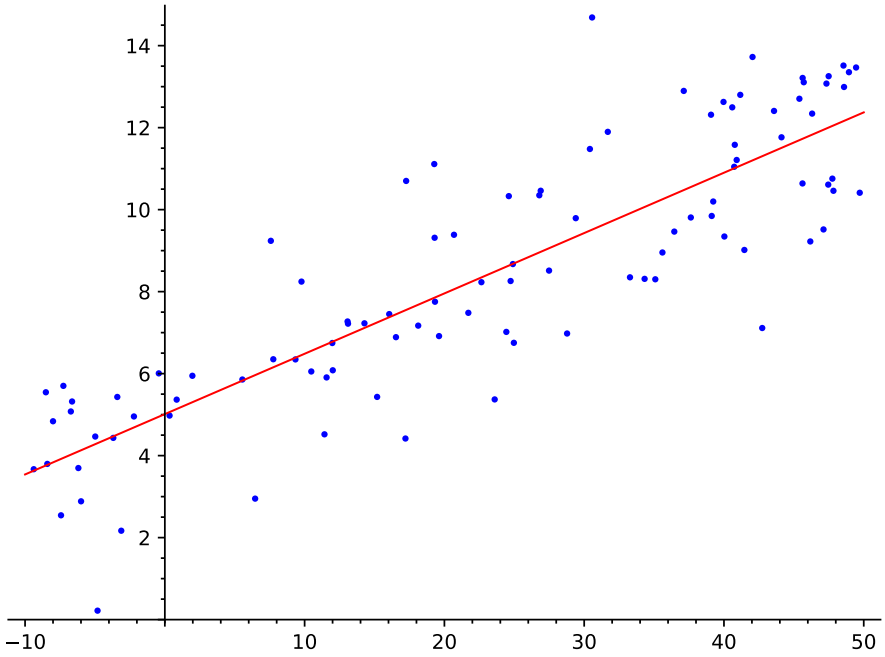

FIGURE 1. Linear Regression applied to a Dataset by using Least squares algorithm.

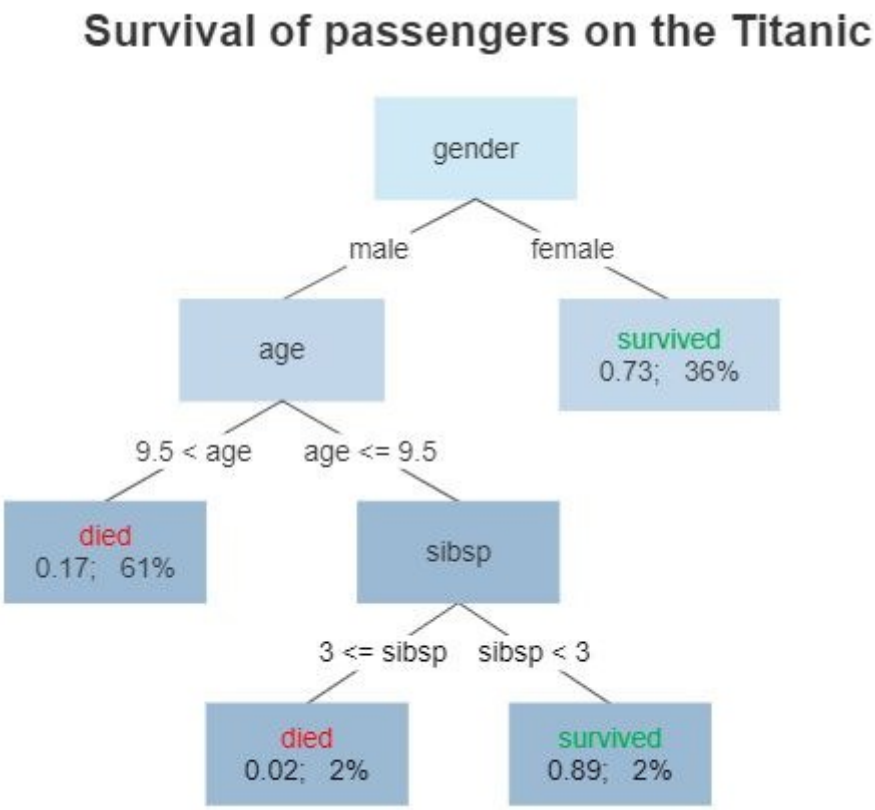

FIGURE 2. Decision Tree Model illustrating the Survival Probability of passengers on the Titanic. sibsp = Number of Siblings/Spouses Aboard. [9].

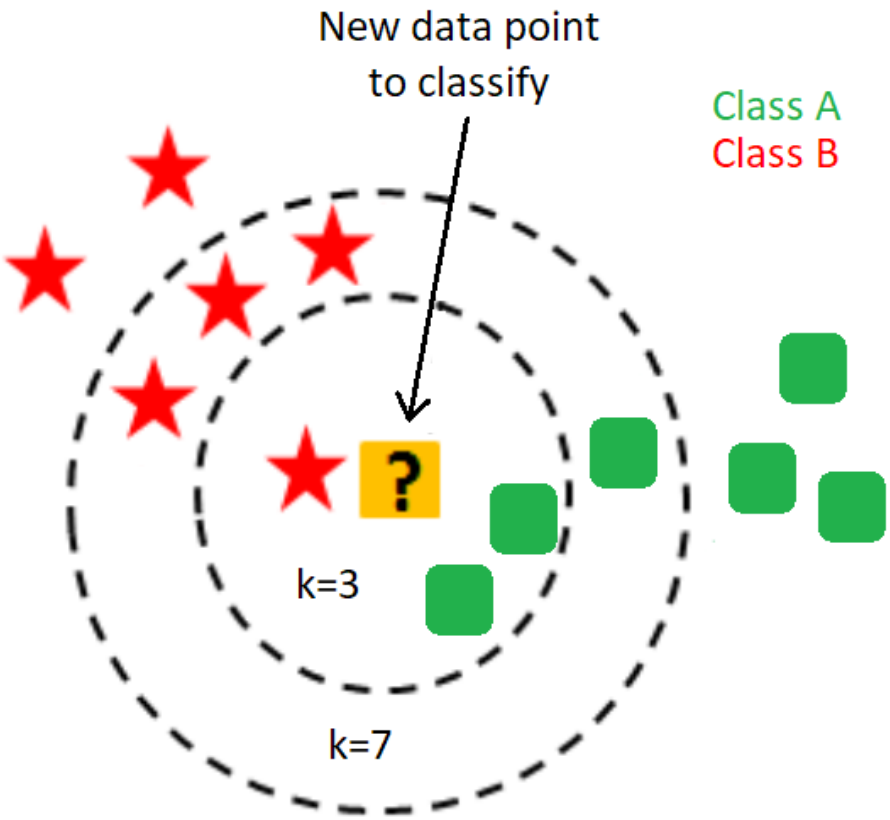

FIGURE 3. Classification of a new data point using K-nearest neighbors algorithm. When k=3, the new data point is classified as A. When k=7, the new data point is classified as B.

2.1.3 Machine Learning algorithms

Machine learning can be divided into two primary classifications for algorithms: unsupervised or supervised learning. These classifications are based on the process the algorithms undergo during their learning phase. [7].

Unsupervised learning algorithms, on one hand, encounter a dataset abundant in features, then gain insights about significant elements of its structure. In deep learning, the usual objective is to understand the full probability distribution from a given dataset, either explicitly, as seen in density estimation, or implicitly, for functions such as synthesis or noise reduction. Some unsupervised learning algorithms also serve different purposes, like clustering, which involves segmenting the dataset into groups of similar examples. [7].

Supervised learning algorithms [7] process a dataset full of features, but each individual also carries a label or target. For instance, email spam detection based on a supervised learning algorithm will learn from an email dataset with “spam” or “not spam” labels and eventually identify spam emails based on features like sender address, content, or subject line.

In addition to supervised and unsupervised learning, there are other types of machine learning algorithms that have emerged. For example, Semi-supervised learning uses the combination of labeled and unlabeled datasets during the training period, making it lies between supervised and unsupervised learning [10]. Reinforcement learning algorithms work by putting AI agents in a feedback environment and making them automatically learn from the experience and improve their performance over time [11]. Dimensionality reduction reduces the number of random variables in a dataset by finding a smaller set of principal variables that capture the most information, either by eliminating or extracting features [12]. Self-learning is a form of learning that does not rely on external rewards or teacher guidance, but rather on the agent's own intrinsic motivation [13]. Feature learning aims to discover better representations of the input data during training, transforming it in a way that makes it more suitable for other tasks [14]. Feature learning can often be used as a pre-processing step before applying other algorithms.

2.1.4 Process of Machine Learning

The process of machine learning typically involves four cycles:

1. Data collection: The first step in machine learning is to collect and prepare a dataset. Preparing the dataset can involve cleaning and formatting the data, selecting relevant features, and may divide the data into training, validation, and testing sets.
2. Training: The machine learning model is trained on the prepared dataset, using algorithms mentioned in the Machine Learning algorithms section. The model is optimized to minimize its error or maximize its accuracy on the problem being solved.
3. Testing: After the model has been trained, it is evaluated on a separate dataset to measure its performance. Hyperparameters can be fine-tuned, that so subsequence training can be improved.

4. Deployment: Once the model has been trained and tested, it can be deployed in a production environment. The model may be integrated into an application, website, or other system. As time passes, new data becomes available, restarting the cycle and improving the model over time.

2.2 Artificial Neural Network

An Artificial Neural Network (ANN) [13] consists of many interconnected units called artificial neurons, inspired by the neurons in a biological brain. Similar to the synapses in a biological brain, each connection can send a signal to other neurons (as illustrated in Figure 4). An artificial neuron takes signals as inputs, processes them, and may produce signals as outputs to other neurons. This signal data type is a real number. The output of each neuron is calculated by applying an activation function to the sum of its inputs. The connection between neurons is named edge, which modifies the intensity of the signal at a connection. Edge weights usually get updated as learning progresses. [15].

Neurons may have a threshold such that they only transmit a signal if the total signal exceeds that threshold. They are being done by activation functions [16] [17] such as sigmoid, hyperbolic tangent (tanh), Rectified linear unit (ReLU), Gaussian, Binary step…

The formula for each node in a basic artificial neural network can be represented as:

$$y = f\left[\sum_{i=1}^{n}(w_i \times x_i) + b\right]$$

where:

- y output value of the node
- f the activation function
- w_i weight for each input x_i
- b the bias value of the node

Neurons in these networks are typically organized into structured layers. Each layer performs unique transformations on its inputs. Signals travel from the first layer, known as the input layer, to the last layer, the output layer, possibly after passing through the layers several times.

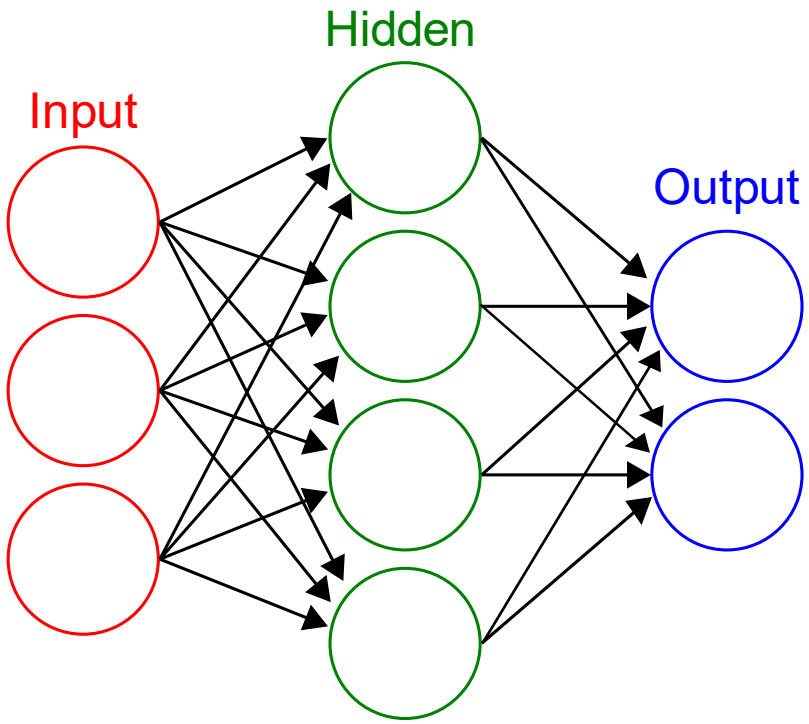

FIGURE 4. Illustration of an artificial neural network with 3 layers.

As mentioned in the Machine Learning models section, one of the ways to improve the generalization ability of the artificial neural network on unseen inputs is to increase its capacity. A common technique to achieve this is to add more layers to the network. Increasing the number of layers can allow the network to learn more complex and abstract features from the data; and achieve better performance and efficiency than shallow networks. [7, p. 163].

Having more layers in the networks also requires understanding how the network learns and operates. Therefore, new research methods are being developed to visualize and analyze ANNs. For example, Zeiler and Fergus [18] proposed a novel visualization technique that gives insight into the intermediate feature layers' function and the classifier's operation (see Figure 5). They also performed an ablation study to measure the performance contribution of different model layers and find model architectures that achieved better results on the ImageNet classification benchmark.

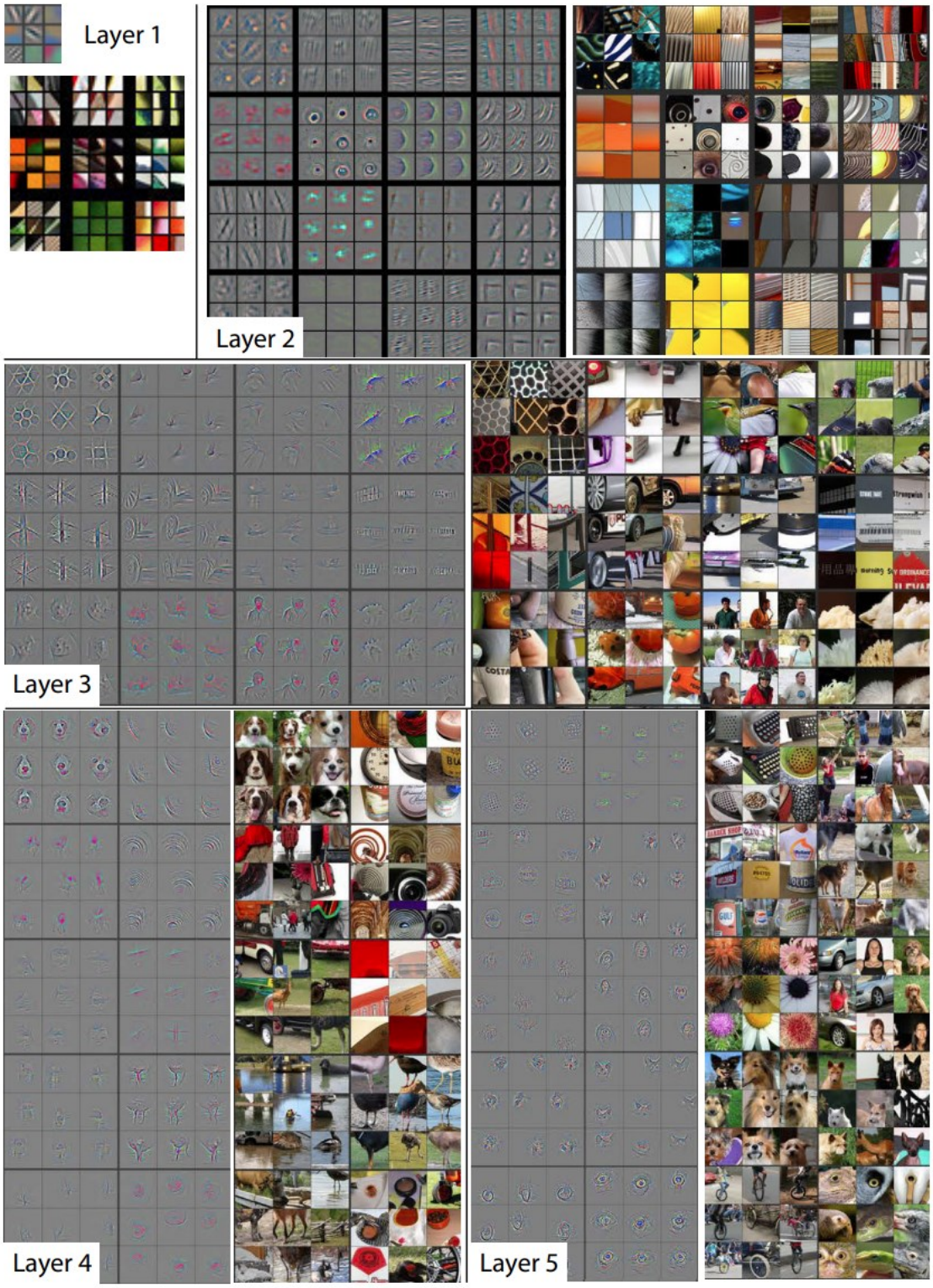

FIGURE 5. A visual representation of the correlation between activated neurons in layers and learned features in a trained model [18].

However, increasing the number of layers also poses some challenges, such as the problem of vanishing gradients and degradation [19] [20]. The vanishing gradient problem commonly arises during gradient descent training in neural networks that have several neuronal layers. In the deeper level of the network, computed gradients are very small or zero, and then little to no training can take place (see Figure 6). In the degradation problem, as the depth of the network increases, model accuracy reaches a saturation point, which might not be surprising, and then starts to decline rapidly. Notably, this decline is not due to overfitting. In fact, adding more layers to an already deep model results in higher training error [21] [22]. Therefore, some techniques are needed to overcome these challenges, generally by adding skip connections [23] such as residual connections [24], batch normalization [25], Multi-level hierarchy [26], Long short-term memory [27], …

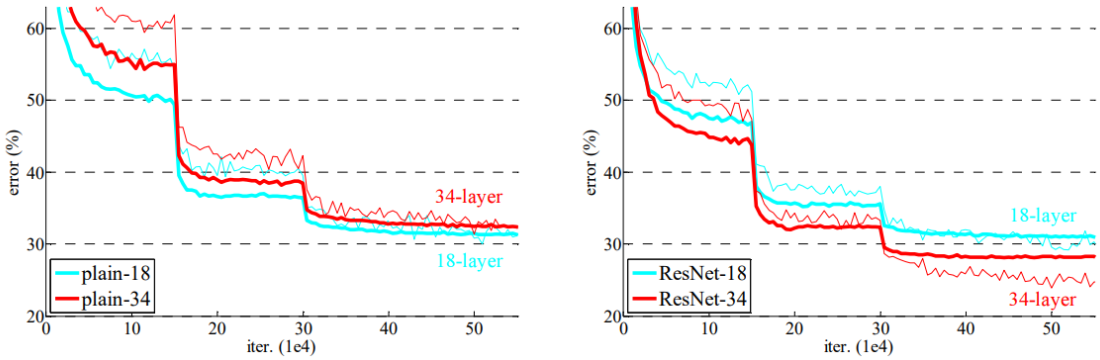

FIGURE 6. Comparison of training errors between plain network and ResNet architecture. The plain network architecture exhibits the vanishing gradients and degradation problem, where the deeper network (34 layers) does not show improvement over the shallower network (18 layers). [24].

2.3 Deep Learning

Deep Learning, a subset of the broader field of machine learning, leverages the principles of artificial neural networks to learn patterns directly from data [28]. The adjective "deep" denotes the multi-layered structure of these networks, allowing them to model complex relationships in the data [7]. Several distinct attributes characterize deep learning:

- Artificial neural networks: These form the basic units of deep learning models.

- Multiple Layers: Deep learning models' capability to harness numerous layers of artificial neurons for data processing sets them apart. This multi-layer approach enables the model to uncover sophisticated data representations and identify patterns that might escape human detection.
- Massive computational resources: Deep learning requires a lot of computing power, which has led to the development of specialized hardware, such as GPUs and TPUs, that can perform the complex calculations required by these models.

Deep learning stands as a robust tool that has empowered machines to learn and model intricate relationships in data. With the continuous evolution of this field, we anticipate emerging applications and methodologies enabling machines to learn and adapt more proficiently than ever before.

2.4 Image processing Neural Network models

Plain simple neural networks can serve as the basis for training ML models. However, not all neural networks are equally efficient regarding training and running speed. Some experiments and research publications [29] have demonstrated in Figure 7 that certain neural network families perform better than others).

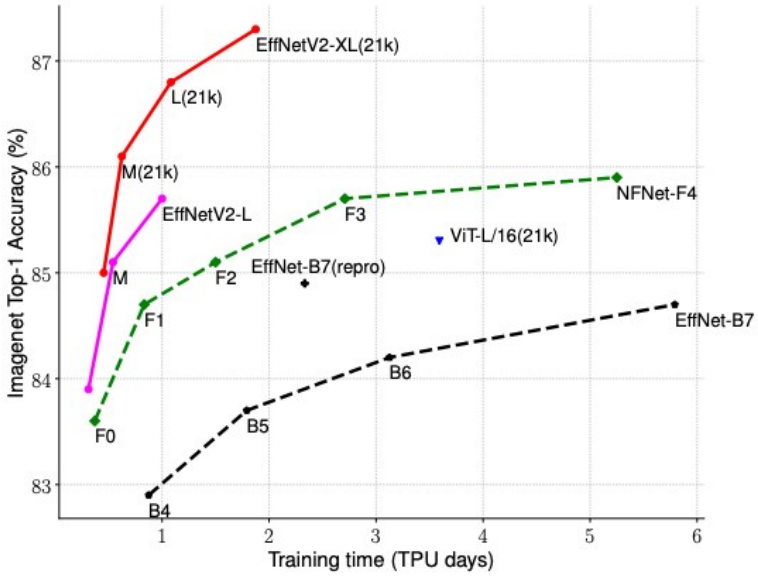

FIGURE 7. EfficientNetV2 surpasses previous models in terms of efficiency for ImageNet classification [30].

In this thesis, we will focus on ANN families within convolutional neural networks, which is a type of neural network that uses convolutional layers to extract features from images. This kind of network is very effective for machine learning tasks involving image processing.

2.4.1 Convolutional Neural Network

Convolutional neural network (CNN) [31] is an artificial neural network that can process images and other types of data with a grid-like structure. A convolution network consists of one or more convolutional layers, followed by activation functions layers and optional pooling layers (see Figure 8). A convolutional layer applies a set of filters to the input data, producing a set of feature maps that capture local patterns in the data. An activation function layer introduces nonlinearity to the network, allowing it to learn complex functions. A pooling layer reduces the size and complexity of the feature maps by applying a down sampling operation, such as max-pooling or average-pooling. A convolution network can learn to extract high-level features from the data and perform tasks such as image classification, object detection, face recognition, and more. [32].

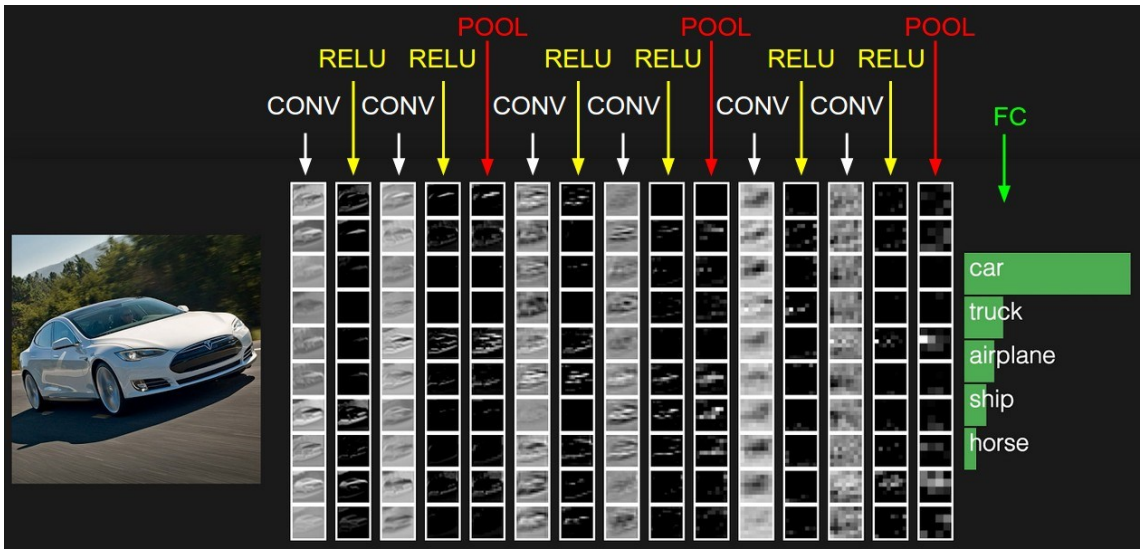

FIGURE 8. The activations of a sample ConvNet architecture are depicted with activation for each layer [33].

2.4.2 ResNet

Residual Network is an innovative deep convolutional neural network widely used in computer vision tasks. Using residual connections allows the network to learn residual functions instead of direct mappings. This makes it easier for the network to learn very deep architectures, which can improve its accuracy and performance. [24].

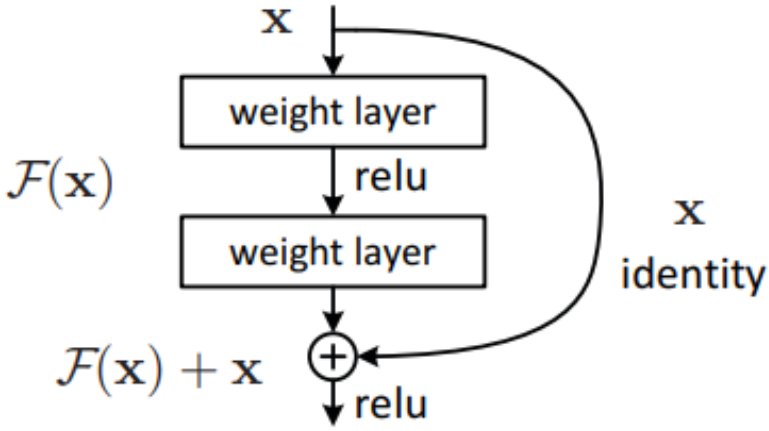

FIGURE 9. A building block of Residual Network [24].

ResNet architecture employs a highly structured organization composed of several distinct stages. Each stage houses a sequence of residual blocks, as demonstrated in Figure 9, with each block encompassing multiple convolutional layers. The ResNet architecture uses 3x3 convolutional filters throughout the network, with occasional 1x1 filters used to reduce the number of dimensions between the 3x3 layers. The convolutional layers function symbiotically with residual connections or 'shortcuts', enabling information to bypass certain layers within the blocks, thus facilitating learning of residual functions. [24].

Residual connections make it easier to train very deep neural networks, because they help to address the problem of vanishing gradients. An overview of Residual connections within the network can be viewed in Figure 10, where there is clear distinct differences between Residual network and Plain network. The architecture also includes batch normalization layers after each convolutional layer, which helps stabilize the training process and improve the model's accuracy [24].

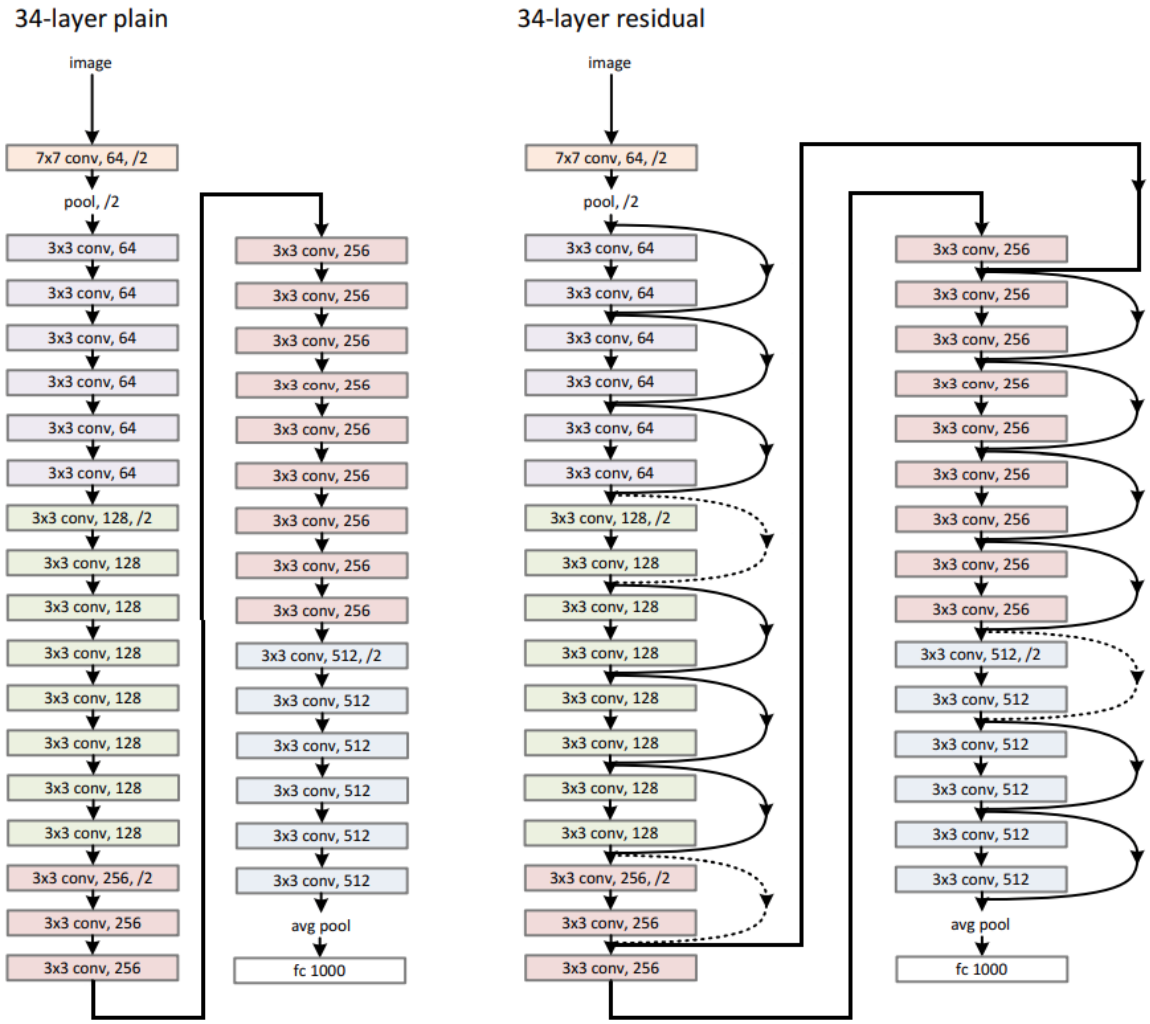

FIGURE 10. Example network architectures. Left: a plain network with 34 layers architecture. Right: a residual network with 34 layers architecture.

Within the ResNet family, there are several variants, including ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152. The number in the name corresponds to the number of layers in the network. For example, ResNet-50 has 50 layers, while ResNet-152 has 152 layers.

2.4.3 MobileNet

MobileNet is a deep convolutional neural network optimized for running on mobile and embedded devices. MobileNet uses depth-wise separable convolutions, which reduce the network's required computations while maintaining its accuracy. In a depth-wise separable convolution, the standard

convolution operation is split into two separate operations (see Figure 11). The depth-wise convolution first applies a single filter to each input channel separately, creating a set of intermediate feature maps. The pointwise convolution then applies a 1x1 convolution to combine these intermediate feature maps into the output. [34].

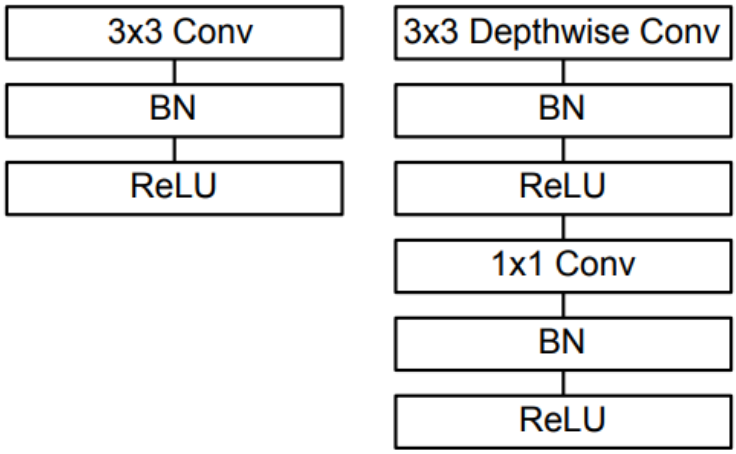

FIGURE 11. Standard convolution layer on the left side and Depth-wise followed by Point-wise layer on the right side [34].

By using depth-wise separable convolutions, MobileNet can achieve high accuracy with a much smaller number of parameters than traditional convolutional neural networks. This makes it well-suited for running on mobile and embedded devices with limited computing resources.

2.4.4 VGGNet

VGGNet is a deep Convolutional Neural Network architecture with multiple layers, developed by the Visual Geometry Group. The number of weight layers varies from 16 to 19, depending on the version of VGGNet (Figure 12 shows the visualization of VGGNet-16). The VGG group achieved remarkable results in the ImageNet Challenge 2014, winning first place in the localization task and second place in the classification task. The VGGNet architecture is also a foundation for many other object recognition models, as it outperforms baselines on various tasks and datasets beyond ImageNet. Furthermore, it was one of the most widely used image recognition architectures during that time. [35].

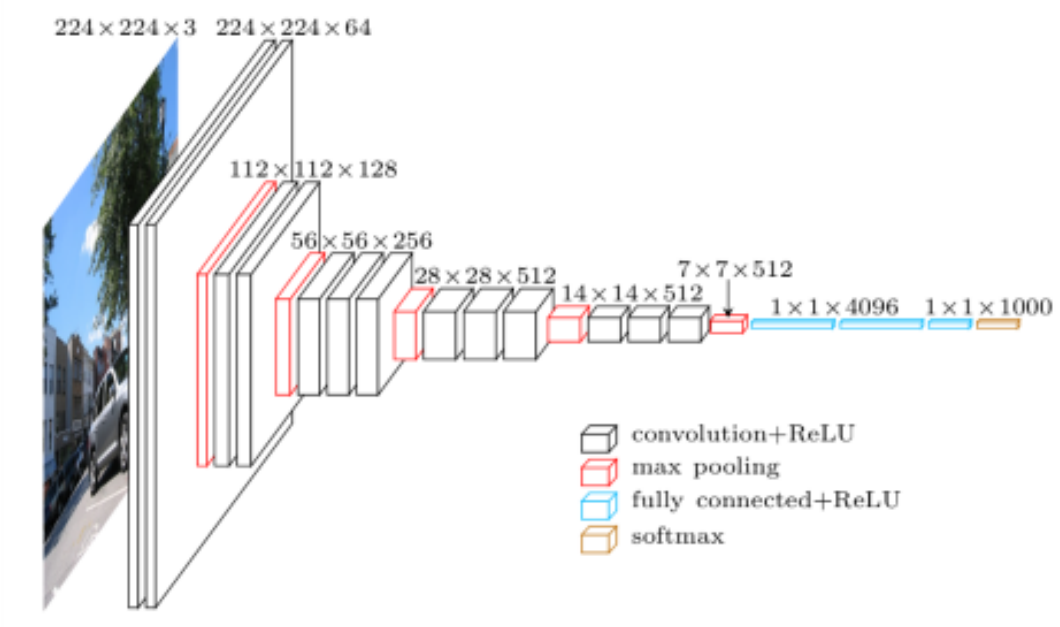

FIGURE 12. Visualization of VGGNet-16 architecture [36].

2.4.5 Pooling-based Vision Transformer

The Pooling-based Vision Transformer (PiT) is a specialized deep neural network designed for tackling image processing tasks, notably image classification and object detection. The distinctive innovation in PiT lies in its implementation of patch-wise processing, which offers a more efficient way to process large images. PiT divides the input image into a grid of patches, each processed individually by the network. This approach of handling smaller, separate images greatly reduces the computational demands and memory requirements. [37].

PiT uses a transformer-based architecture, like the one used in natural language processing tasks such as machine translation and text generation. The transformer architecture is based on the use of self-attention mechanisms, which allow the network to learn dependencies between different parts of the input. By using patch-wise processing and the transformer architecture, PiT can achieve significant performance on various image classification tasks, while requiring fewer computations and parameters than other deep neural network architectures. [37].

2.5 Transfer Learning

A pretrained network is a ready-made machine learning model that another developer has trained on a large and rich dataset, usually for a specific and complex task. A pretrained network can be used as a starting point for a learning model in a second learning task [38]. This allows rapid progress when modeling the second task [39]. As a result, transfer learning saves time and resources, as well as improves the performance of the new model, especially when the new task is similar to the original one or the new data is scarce or limited.

2.6 Measurement

The performance of a machine learning model can be measured using various metrics, depending on the specific task being performed:

- Accuracy: Accuracy is the most used metric for classification tasks. It measures the percentage of correct predictions made by the model. However, accuracy can be misleading if there are imbalanced classes, or the cost of false positives and false negatives is different.
- Precision and recall: precision gauges the percentage of the model's positive predictions that are indeed correct. Recall assesses the proportion of actual positive instances the model managed to capture.
- F1 score [40]: The F1 score is the harmonic mean of precision and recall and is often used as a balanced measure of a model's performance. It provides a single score that considers both precision and recall.
- Mean squared error (MSE): measures the average squared difference between the predicted and actual values.
- Root mean squared error (RMSE): RMSE is the square root of the MSE and is often used as a more interpretable measure of a model's performance in regression tasks.
- Area under the receiver operating characteristic curve (AUC-ROC) [41]: AUC-ROC is a metric used to evaluate binary classification tasks. It measures the model's performance across different thresholds and provides a single score considering sensitivity and specificity.

3 ENGINEERING ANALYSIS

3.1 Machine Learning model

The choice of a deep neural network architecture, such as ResNet, MobileNet, VGGNet, or PiT, depends on the specific requirements of the task at hand. Particularly regarding accuracy, the primary consideration for most applications is accuracy. ResNet is a well-known architecture often used for image classification tasks, especially when a high level of accuracy is required. MobileNet is optimized for running on mobile and embedded devices, and has fewer parameters than ResNet, making it faster and more efficient. VGGNet was a successful architecture in the past, but the VGG group has not updated it and has fallen behind in performance. It suffers from the vanishing gradient problem, which makes it hard to train deeper models. PiT is a relatively new architecture that has shown promising results in image classification tasks. Besides, the computational requirements of the architecture are also an important consideration. ResNet is a deep and computationally expensive architecture, while MobileNet and PiT are both lighter and more efficient, making them better suited for low-power devices or real-time applications.

Another consideration is the quality of data. For example, the size and quality of the input images may also influence the choice of architecture. MobileNet and PiT are both designed to handle smaller input images, while ResNet and VGGNet may be better suited for larger or higher quality images. Also, pretrained models are available for many popular deep neural network architectures, which can save time and computational resources. ResNet, MobileNet, and VGGNet are well-established architectures with many pretrained models available. At the same time, PiT is a newer architecture with fewer pretrained models available, especially not available in PyTorch library, making it hard to utilize transfer learning. Utilizing transfer learning is further discussed in the Methodology section.

For those reasons, this work focuses on two different neural network architectures for age verification: ResNet-50 and MobileNet. ResNet-50 is a popular and powerful model that achieved a breakthrough in image processing. MobileNet is a fast and efficient model that can run locally on mobile devices, enhancing the privacy and security of user data in various scenarios.

3.2 Dataset

The usage of datasets plays a crucial role in machine learning. A high-quality dataset can significantly impact the learning process by providing diverse examples and accurately representing the real-world phenomena from which the model can learn [42]. Using popular, well-established datasets can also be beneficial, as they have been tested and used by many researchers and are likely to contain less bias and errors. Examples of how datasets appear can be viewed in Figure 13 and Figure 14.

Three popular datasets with age labels that we are considering are:

- VGG-Face2 Mivia Age Estimation (VMAGE) dataset [43] contains approximately 3 million labels, based on the VGGFace2 dataset [44] contains 3.3 million images with around 9,000 identities.
- IMDB-WIKI dataset [45] [46] contains around 500,000 images with labels and is a trusted source for age estimation projects. Its credibility comes from the data entries typically being uploaded with precise, manually entered information.
- UTKFace aligned & cropped faces dataset contains approximately 23000 images, from the UTKFace dataset [47].

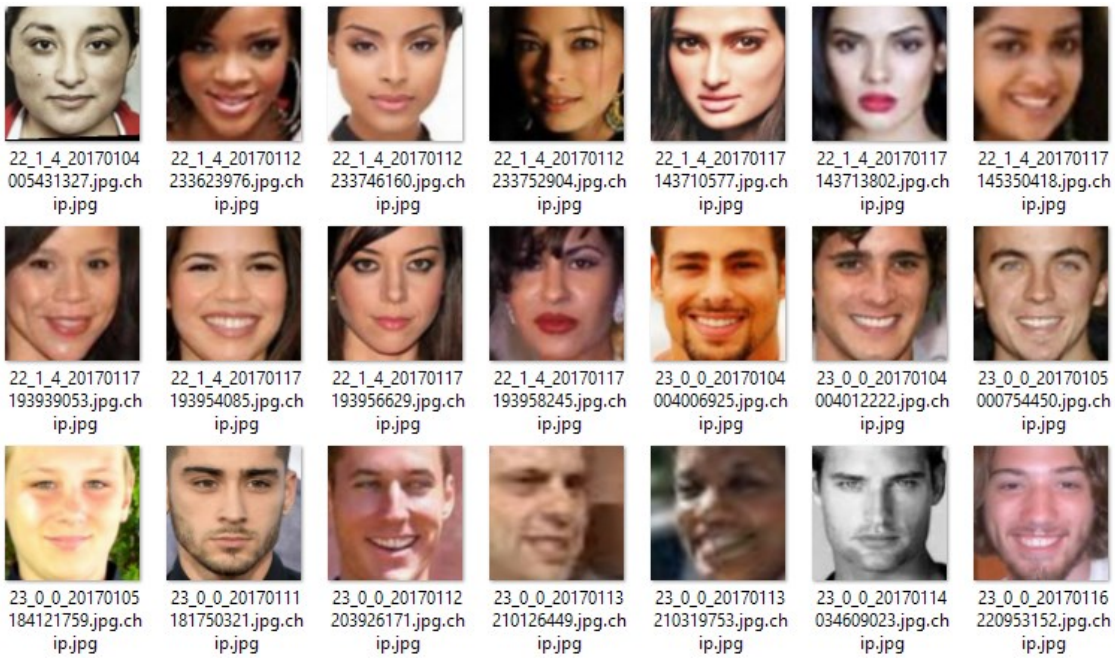

FIGURE 13. Sample images from UTKFace Aligned & Cropped Faces dataset with embedded labels in the filename with format [age]_[gender]_[race]_[date&time].jpg.

train.age_detected.csv

```
825735    n002432/0066_01.jpg,27.579068768507764
825736    n002432/0067_02.jpg,32.211402877477795
825737    n002432/0068_01.jpg,29.600717862663217
825738    n002432/0069_01.jpg,26.44872010053526
825739    n002432/0070_01.jpg,26.309787635068798
825740    n002432/0071_01.jpg,30.9980960174286
825741    n002432/0072_01.jpg,33.04026397989643
825742    n002432/0073_02.jpg,29.929451054553617
825743    n002432/0074_01.jpg,25.971271631268827
825744    n002432/0075_01.jpg,35.67440972428018
825745    n002432/0076_03.jpg,30.699992638500593
825746    n002432/0077_02.jpg,31.140864123358543
825747    n002432/0078_01.jpg,34.921394271166164
825748    n002432/0079_01.jpg,35.04962477354681
```

FIGURE 14. Extracted age annotation from VMAGE Datasets in CSV Format: VGGFace2 image paths are listed in the left column, with corresponding age annotations in the right column.

While the VMAGE dataset is large and may offer a diverse range of examples, its sheer volume of data would require considerable computational resources for processing and training. Hence, the choice was made to utilize the IMDB-WIKI dataset. With a substantial amount of 500k samples, this dataset still offers diversity and sufficient examples for learning, but with a less taxing demand on computational resources compared to VMAGE.

Additionally, the UTKFace dataset, despite its smaller size, will be employed for testing purposes after training the models, providing a diverse and independent set to validate the model's performance.

3.3 Methodology

In training our models, we examine three distinct methodologies.

- Starting from Zero: we begin the learning process without any pre-existing knowledge.
- Training from pretrained networks: we use premade networks and learn from them.
- Using a fully configured network: we use a complete, already built network to perform our tasks.

By using a fully configured network, we must rely on the existing models' availability, quality, and license, and we may not learn much about the training process. Ultimately, no matter which approach we use, we still have to run performance measurements to validate that it is correctly trained and performs well with our dataset. Moreover, even if we save time using a ready-made network, these reasons do not justify us.

On the other hand, training from a pretrained network will only require us to make transfer learning by starting from a popular pretrained network, which is ResNet-50 based on the ImageNet dataset [24]. While requiring more effort, we will get more learning experience in training ML models rather than going through the trouble of searching for the available models that fit our tasks.

With the advancement of technology, cloud computing has enabled access to powerful computing resources, such as GPU and TPU, that make training from scratch feasible and affordable. For example, it is possible to train a ResNet image classifier from scratch with TPUs on the Google Cloud Platform [48]. Training a new scratch model to achieve 93% accuracy can take as little as 2 minutes and cost as low as $7 [49]. However, training from scratch also entails some challenges and trade-offs. We must carefully evaluate the performance measures as we are giving up the accuracy and features of the well-established pretrained network. Moreover, pretrained ResNet-50 is based on ImageNet [50], a dataset with 14 million images. In contrast, the labeled age dataset only contains a much smaller number of samples, such as the VGGFace2 dataset, which contains 3.3 million images with only around 9,000 identities, or the UTKFace dataset, which contains 20,000 labeled data. Therefore, transfer learning will help us inherit a higher base network from ImageNet and improve our results on the performance.

In this thesis, we aim to demonstrate the application of ML in real-world scenarios where we can leverage more powerful models than ResNet-50 that require more extensive training, even by transfer learning. Therefore, we choose the approach of training from a pretrained network.

3.4 Objective

The goal of age verification based on machine learning is to accurately determine a user's age while minimizing the prediction error. Several variables will invariably influence the precision of age estimation. Therefore, it is crucial to define our goal clearly.

Our target is to confine the error within ±5 years in most instances. Ideally, our model should be able to make predictions with an error of only ±2 years in most cases. Let us denote the actual age as A and the predicted age as P. Our goal can be defined with these formulas:

- For 99% of the cases: $|A - P| \leq 5$
- For 75% of the cases: $|A - P| \leq 2$

Where $|A - P|$ denotes the absolute difference between the actual age and the predicted age.

3.5 Performance measure and Loss function

In age verification based on machine learning, the choice of loss function and the train-test procedure are essential components in determining the accuracy and generalizability of the model.

The train-validation-test procedure is used to evaluate the machine learning model's performance. The data is divided into three sets: training set, validation set, and testing set. The training set is used to train the machine learning model. The validation set is used to validate the model during training. The testing set evaluates the model's performance on new data.

The loss function measures the difference between the predicted and actual age values.
We employ Mean squared error (MSE), a common loss function for regression problems, for our age verification problem [47]. The MSE function calculates the average squared difference between the predicted and the actual values as the following formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2$$

where:

- n the total number of data
- x_i, y_i the predicted value and the actual value

In addition to MSE, these performance measurement metrics are also used to assess the model's accuracy to be in line with our objective:

- Accuracy ± 2: Percentage of predictions within two years of actual age.
- Accuracy ± 5: Percentage of predictions within five years of actual age.
- Accuracy 18: Percentage of accuracy in identifying individuals above or below age 18.
- MAE: Mean absolute error, calculate the average absolute difference between the predicted and the actual values as the following formula:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|x_i - y_i|$$

where n, x_i, y_i are defined as in MSE

3.6 Security

Security is essential for real-life applications, as ML models can be easily fooled if they are not protected because they are simple and only specialized on one task. We need to consider two security vectors that a user can compromise our facial recognition system:

- The first vector is client-side manipulation, which involves tampering with the output of the deployed models on the client device or altering the data during transmission to the backend, as in a traditional hijacking attack.
- The second vector is attacks that fool facial recognition algorithms, such as using printed images, video playing on tablets, and printed face masks [51].

Besides these common attack vectors, more advanced and sophisticated methods can compromise our networks, such as embedding backdoors in pretrained networks or poisoned datasets, extracting sensitive information from our model using model inversion techniques, and more [52] [53]. Some attacks combine both client-side and algorithmic manipulation to create more potent threats, such as using a virtual camera or manipulating physical camera output, adding noise [54], or deepfaking the user's face [55].

In this thesis, we utilize the popular and well-trusted ResNet-50 pretrained network, thereby mitigating potential issues related to embedded backdoors. However, the focus of this thesis does not include resolving these security issues.

3.7 Technology selection

3.7.1 Programming language for Machine Learning development

We evaluated the following programming languages for writing code to train our models: Python, NodeJS, and C++. Our preferred option: Python with PyTorch: We selected PyTorch as our machine learning framework based on Python because of its popularity, wide usage in AI development, and strong support on various cloud platforms for future work.

Other possible options:

- NodeJS is a viable and well-supported language with similar advantages and disadvantages to Python. However, it has a smaller and less mature ecosystem of ML libraries and frameworks than Python, making it less suitable for our project unless we prioritize JS based application over other factors.
- C++ is powerful and complex, providing much higher performance than Python, but with a drawback of long development time that makes it impractical for our project.

3.7.2 Cloud provider comparison for training

We compare two options for setting up a virtual machine (VM) to train our ML models on the cloud. The first option is to use a traditional VM and install our software, such as Python, ML libraries, Jupyter Notebook... This option gives us more flexibility and control over the environment and allows us to reproduce this work on any cloud provider.

The second option is to use a ready-made cloud solution, such as GCP VertexAI or AWS SageMaker. These solutions are automated platforms that provide pre-configured tools and frameworks for ML development and deployment on popular public clouds. With these solutions, we can start coding and training our models right after the VM is created. They also offer features such as auto-scaling and easier migration at the hardware and software levels. However, they have some drawbacks, such as higher training costs (Table 1 demonstrates 20-40% more than traditional VMs). These solutions might be more suitable for quick or small-scale development, as they can eliminate some DevOps tasks at a relatively low cost.

TABLE 1. Price Comparison between AWS SageMaker [56] and AWS EC2 [57] in Stockholm (eu-north-1).

SM Instance Type	SM Cost	EC2 Instance Type	EC2 Cost	SM/EC2 Cost Ratio
ml.c5.large	0.109	c5.large	0.091	1.2
ml.c5.xlarge	0.218	c5.xlarge	0.182	1.2
ml.c5.4xlarge	0.874	c5.4xlarge	0.728	1.2
ml.c5.24xlarge	5.242	c5.24xlarge	4.368	1.2
ml.m5.large	0.122	m5.large	0.102	1.2
ml.m5.xlarge	0.245	m5.xlarge	0.204	1.2
ml.m5.4xlarge	0.979	m5.4xlarge	0.816	1.2
ml.m5.24xlarge	5.875	m5.24xlarge	4.896	1.2
ml.g4dn.xlarge	0.781	g4dn.xlarge	0.558	1.4
ml.g4dn.2xlarge	0.998	g4dn.2xlarge	0.798	1.25
ml.g4dn.4xlarge	1.596	g4dn.4xlarge	1.277	1.25
ml.g4dn.8xlarge	2.885	g4dn.8xlarge	2.308	1.25
ml.g4dn.16xlarge	5.771	g4dn.16xlarge	4.617	1.25
			Average	1.23

4 IMPLEMENTATION AND RESULT

4.1 ML Training

4.1.1 Technology and Hardware requirements

- Instance with 2 CPUs, 8 GB memory, and 1 NVIDIA V100 GPU.
- Python / PyTorch.

4.1.2 Training prerequisites

Before training the model, ensur recision and recalle all necessary libraries are installed. The required libraries include: libgl1, cmake, build-essential, numpy, opencv-python, pandas, matplotlib, scikit-learn, easydict, dlib, torch, torchvision, and torchaudio.

Moreover, for the utilization of a GPU during the training process, the installation of the GPU driver is necessary. For example, NVIDIA GPUs require the CUDA driver.

The dataset also needs to be downloaded. Instructions for locating and downloading the dataset are provided in the dataset sources.

4.1.3 Training algorithm

We divide the main training algorithm into three steps:

- Load the model from torchvision library [58] and load all previous training data if needed.

```
from torchvision import models

...
model = models.resnet50(weights=models.ResNet50_Weights.DEFAULT)

...
# if we are loading from an earlier training
state = torch.load("../data/training_models/LastModel.pth")
model.load_state_dict(state_dict)
```

- Split the original training data into two sets: training data and validation data, to assess the performance of the models during training. We can randomly partition the datasets by using `train_test_split` from `sklearn.model_selection` [59]. By default, `train_test_split` will use 25% dataset for validating and 75% for training [60].

```
from sklearn.model_selection import train_test_split

...
indices = np.arange(0, dataset_length)
indices_train, indices_eval = train_test_split(indices)

# Load the dataset (image and label) to a mapped dictionary
train_data = DatasetMapped(Cfg.dataDir, label_path, indices_train)
eval_data = DatasetMapped(Cfg.dataDir, label_path, indices_eval)

# Load the mapped dictionary to DataLoader, which will be used to create
batches for training
train_loader = DataLoader(train_data, batch_size=4, shuffle=True,
num_workers=2)
eval_loader = DataLoader(eval_data, batch_size=4, shuffle=True,
num_workers=2)
```

- Finally, we train our model. Each training step is called a training epoch. In each, we perform a train-validation procedure to calculate training loss, validation loss, and accuracy. We save the latest model after each training epoch and the best model with the lowest validation loss. The Performance measure and Loss function section defines the calculation for validation loss.

```
# criterion are defined loss function
def validation_epoch(...):
    for i, (images, age_truth) in enumerate(eval_loader):
        images = Variable(images)
        age_predict = model(images)
        validation_loss = criterion(age_predict, age_truth)


def train(...):
    ...
    train_epoch(model, train_loader, criterion, optimizer, epoch)
    validation_loss = validation_epoch(model, validation_loader,
criterion)
    if validation_loss <= best_loss:
        ...
        torch.save(state, "../data/training_models/BestModel.pth")

    ...
    torch.save(state, "./training_models/Last_Model.pth")
```

4.1.4 Testing

Once the training phase is complete, the next step involves testing the model to quantify its performance. In this phase, the model's accuracy and testing loss is calculated using the images from the UTKFace dataset as the testing dataset.

```
...
model = load_model()
model.eval()  # set the model to evaluation mode

...
for i in range(n):
    ...
    file_name = data[0][i]
    ground_truth = data[1][i]

    ...
    # process img
    res = model(img)
    val = round(res[0].item())

    ...
    ground_truth_values[i] = data[1][i]
    predict_values[i] = val

...
# calculate accuracy data, such as MSE or counting accuracy
MSE_val = cal_MSE(predict_values, ground_truth_values)
accuracy_val = cal_accuracy(predict_values, ground_truth_values)

print(' MSE_val: ', MSE_val)
print(' accuracy: ', accuracy_val)
```

4.1.5 Result

The outcome is presented in Table 2.

TABLE 2. Result of the training.

Model	Total Epoch	MAE	MSE	Accuracy ± 5 (%)	Accuracy ± 2 (%)	Accuracy 18 (%)
ResNet-50	45	6.12	73.72	58.77	31.99	94.56
ResNet-50	52	6.13	74.71	59.10	31.70	95.03
MobileNet	55	6.09	72.48	59.50	31.81	95.24

The metrics for accuracy ± 5 and ± 2 have yielded much lower results than our target. Additionally, as training progresses, the MAE and MSE metrics do not show signs of reduction. These pre-

liminary results prompt a deeper investigation. It is essential to note that this does not necessarily indicate a failure of the approach itself, but rather suggests potential issues to be examined in the following section.

4.2 Analyzing result

The training and validation loss data analysis revealed a notable discrepancy (see Figure 15), indicative of a possible problem. Since both training and validation employ the MSE as a loss function, their relative values should ideally be low. However, observation indicates that while the training loss is low, the validation loss remains high and does not show a substantial decrease. This clearly indicates that our model is memorizing the training data rather than learning to generalize from it. Detailed figures illustrating this issue can be found in Appendix 1.

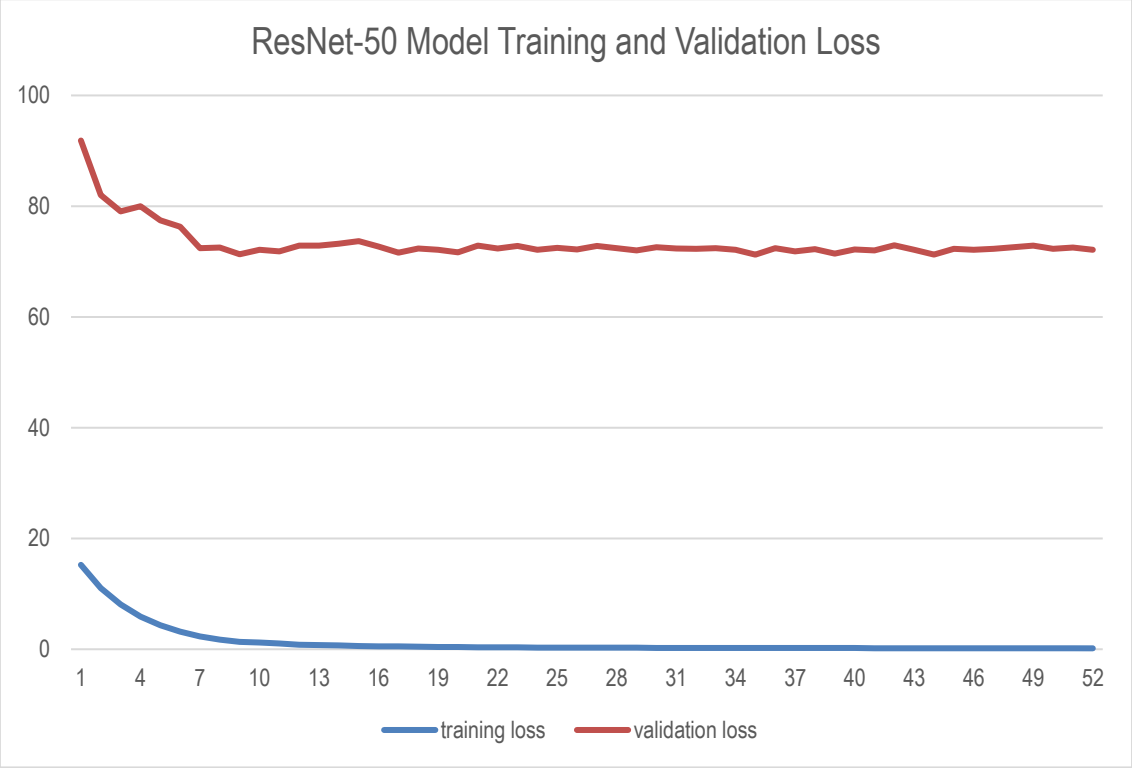

FIGURE 15. Comparison of Training and Validation Loss for ResNet-50 model. The analysis reveals a significant discrepancy between the two. While the training loss shows a low value, the validation loss remains high and does not exhibit a substantial decrease.

Upon reviewing the algorithm in light of the overfitting issue, it became evident that additional measures must be implemented to mitigate this problem. Several diagnostics have been done with their proposed solutions:

- Data augmentation: involves making meaningful modifications to our data to be more aligned. Although we have precautioned pick face aligned & cropped datasets, more work needs to be done regarding background removal. This can be achieved either by employing a face detection algorithm for comprehensive background elimination, or subtracting the `mean_background` data, which can be calculated from the training dataset. The latter approach assumes that a good dataset would reflect real-world scenarios, thus justifying the reuse of this `mean_background` value.
- Dropout: It was noted that the Dropout function from PyTorch was not utilized in the present implementation. This can be introduced by invoking `nn.Dropout()` and subsequently integrating it into the feedforward function. The Dropout technique [61] randomly zeroes some elements of the input tensor during training with a probability. Each channel is independently zeroed out on every forward call, preventing co-adaptation of neurons, and hence reducing overfitting.
- Weight decay: Weight decay is a regularization technique that prevents overfitting by discouraging complex models, specifically by keeping the weights small. It does this by adding a penalty term to the loss function, which is a function of the magnitude of the weights. The weight decay value determines the strength of the regularization. This can be done by passing `weight_decay` when creating a `torch.optim.SGD` object

After looking at related studies on age estimation, it was reassuring to find that our project goals are indeed realistic. For instance, the study conducted by Bao et al. [62] achieved the MAE value of 1.86 and a standard deviation value of 0.2. This indicates that our target performance metrics are within a feasible range. Notably, their research utilized a subset of the VMAGE dataset, approximately 500,000 images in size, which aligns with our chosen dataset size. Given that both the VMAGE and IMDB-WIKI datasets are widely used in the research community, it is inferred that our dataset is both suitable and sufficiently diverse for the given task.

Furthermore, the study began with an interesting approach of using ResNet-18 as a baseline model, and later transitioned to using EfficientNetV2-M for better performance and training effi-

ciency. This choice of a simpler model for the baseline can aid in avoiding noise detection and thus promote better generalization. However, we believe that with proper regularization techniques, sufficient data, and fine-tuning of the learning rate or other hyperparameters, a deeper network like ResNet50 or ResNet152 might perform well. Future exploration could potentially involve other architectures such as EfficientNet, DenseNet, or even transformer-based models like Vision Transformer (ViT).

5 CONCLUSION AND FURTHER DEVELOPMENT

5.1 Conclusion

The primary goal of this thesis was to devise a machine learning based approach for user age verification from facial images, thereby reducing reliance on sensitive identification data. Despite diligent efforts and rigorous evaluations using multiple performance metrics, the model fell short of its expected accuracy due to overfitting issues.

The lessons learned from this study underscore the importance of data quality, effective regularization techniques, and model robustness to avoid overfitting and improve predictive accuracy. While stemming from the specific challenge of age verification, these insights can be universally applied across diverse machine learning applications, particularly those involving image-based prediction tasks.

Although the model did not reach its intended performance, the methodology and foundational understanding developed in this thesis could serve as a resource for practitioners aiming to integrate machine learning based age verification solutions. The analysis and proposed mitigation strategies for overfitting can guide further enhancements to improve model performance.

5.2 Unresolved issues and Future works

5.2.1 Overfitting issue and mitigation strategies

The primary issue encountered in this thesis was overfitting, which compromised the model's performance. Upon analysis, it became clear that refining our approach to data augmentation, applying dropout techniques, and implementing weight decay as a regularization strategy could help mitigate the overfitting issue and improve model performance. This should be our main priority before proceeding to explore other approaches.

5.2.2 ML Model security challenges

Machine learning models, including age verification technologies, constantly face new attack vectors that challenge security and reliability. However, this does not mean they are not worth pursuing and improving. Like all automated solutions, they are vulnerable to security attacks but also present opportunities for innovation and progress. For a higher level of security or identity verification, such as protecting sensitive or valuable information, ML technologies may not be enough, and require stronger and more reliable authentication methods, such as OAuth.

A notable challenge is the use of makeup that can significantly alter facial features, potentially fooling age verification models. Historically, makeup has been used to make faces appear younger or even impersonate another individual. Given its capacity to visually deceive even human observers, makeup indeed poses a significant challenge for machine learning models. This presents an interesting area for future work - enhancing the ability of machine learning models - especially those employed for age estimation and facial recognition - to learn from, adapt to, and adjust predictions when alterations such as makeup are present in the data.

In the future, we will explore new defense solutions to mitigate potential vulnerabilities in our facial recognition system against sophisticated threats. These threats include backdoors in pretrained networks, poisoned datasets, and model inversion techniques. The continual evolution of the system's robustness in the face of such challenges will be a critical aspect of future research.

5.2.3 Explore other datasets

Due to limited resources, the IMDB-WIKI dataset was chosen for training during this project. However, larger datasets like VMAGE may potentially offer improved results. In the future, with access to greater computational resources, it would be beneficial to explore the effects of training with these larger datasets. This could lead to more diverse learning examples and potentially enhance the accuracy of age estimation models.

5.2.4 More advanced artificial neural networks

In the field of Artificial neural networks, there are several advanced models that consistently rank at the top of leaderboards for image classification problems, as seen on resources like Self-Supervised Image Classification on ImageNet on Papers with Code Leaderboard [63]. Models such as EfficientNet, ResNet-152, and Vision Transformer have shown promise in various classification tasks. Given their superior performance in these areas, they could potentially improve the accuracy and efficiency of age estimation. Therefore, future research could explore the application of these state-of-the-art models to age estimation tasks, potentially leading to enhanced results and driving further advancements in this domain.

5.2.5 Parallel and Distribution training

During the training of the age estimation models, it became clear that the process could benefit significantly from parallel and distributed training. This technique speeds up learning by sharing computation across multiple resources. It also allows for gradual scaling: starting on a small computing instance to identify potential issues, then allocating more resources as training proceeds. Therefore, focusing on distributed training could be an advantageous path for future work in this area, promising increased efficiency in training age estimation models.

REFERENCES

[1] C. M. Bishop and N. M. Nasrabadi, Pattern recognition and machine learning, vol. 4, Springer, 2006.

[2] T. Mitchell, Machine Learning. McGraw Hill, 1997.

[3] R. O. Duda and P. E. Hart, *Pattern recognition and scene analysis,* Wiley, New York, 1973.

[4] A. Scriven, D. J. Kedziora, K. Musial and B. Gabrys, "The Technological Emergence of AutoML: A Survey of Performant Software and Applications in the Context of Industry," *arXiv preprint arXiv:2211.04148,* 2022.

[5] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean and W. Fedus, "Emergent Abilities of Large Language Models," *Transactions on Machine Learning Research,* 2022.

[6] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg and others, "Sparks of artificial general intelligence: Early experiments with gpt-4," *arXiv preprint arXiv:2303.12712,* 2023.

[7] I. Goodfellow, Y. Bengio and A. Courville, Deep learning, MIT press, 2016.

[8] S. J. Russell, Artificial intelligence a modern approach, Pearson Education, Inc., 2010.

[9] Gilgoldm, "File:Decision tree.Jpg," 18 May 2020. [Online]. Available: https://commons.wikimedia.org/wiki/File:Decision_Tree.jpg. [Accessed 20 May 2023].

[10] O. Chapelle, B. Scholkopf and A. Zien, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]," *IEEE Transactions on Neural Networks,* vol. 20, p. 542–542, 2009.

[11] L. P. Kaelbling, M. L. Littman and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research,* vol. 4, p. 237–285, 1996.

[12] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science,* vol. 290, p. 2323–2326, 2000.

[13] S. Bozinovski and others, "A self-learning system using secondary reinforcement," *Cybernetics and Systems Research,* p. 397–402, 1982.

[14] Y. Bengio, A. Courville and P. Vincent, "Representation learning: A review and new

perspectives," *IEEE transactions on pattern analysis and machine intelligence,* vol. 35, p. 1798–1828, 2013.

[15] Y. Bengio and others, "Learning deep architectures for AI," *Foundations and trends® in Machine Learning,* vol. 2, p. 1–127, 2009.

[16] J. Lederer, "Activation functions in artificial neural networks: A systematic overview," *arXiv preprint arXiv:2101.09957,* 2021.

[17] C. Nwankpa, W. Ijomah, A. Gachagan and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378,* 2018.

[18] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, 2014.

[19] Y. Bengio, P. Simard and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks,* vol. 5, p. 157–166, 1994.

[20] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010.

[21] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[22] R. K. Srivastava, K. Greff and J. Schmidhuber, "Highway networks," *arXiv preprint arXiv:1505.00387,* 2015.

[23] F. Liu, X. Ren, Z. Zhang, X. Sun and Y. Zou, "Rethinking skip connection with layer normalization in transformers and resnets," *arXiv preprint arXiv:2105.07205,* 2021.

[24] 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.

[25] J. Yang, L. Sang and D. Cremers, "Dive into Layers: Neural Network Capacity Bounding using Algebraic Geometry," *arXiv preprint arXiv:2109.01461,* 2021.

[26] J. Schmidhuber, "Learning complex, extended sequences using the principle of history compression," *Neural Computation,* vol. 4, p. 234–242, 1992.

[27] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation,* vol. 9, pp. 1735-1780, November 1997.

[28] L. Deng, D. Yu and others, "Deep learning: methods and applications," *Foundations and*

trends® in signal processing, vol. 7, p. 197–387, 2014.

[29] A. Kadra, M. Lindauer, F. Hutter and J. Grabocka, "Well-tuned simple nets excel on tabular datasets," *Advances in neural information processing systems,* vol. 34, p. 23928–23941, 2021.

[30] M. Tan and Z. Dai, "Toward fast and accurate neural networks for image recognition," 16 September 2021. [Online]. Available: https://ai.googleblog.com/2021/09/toward-fast-and-accurate-neural.html. [Accessed 20 May 2023].

[31] Y. LeCun and others, "Generalization and network design strategies," *Connectionism in perspective,* vol. 19, p. 18, 1989.

[32] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, p. 2278–2324, 1998.

[33] F.-F. Li, Y. Li and R. Gao, "Convolutional Neural Networks for Visual Recognition," [Online]. Available: https://cs231n.github.io/convolutional-networks/. [Accessed 20 May 2023].

[34] 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.

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

[36] L. Blier, "A brief report of the Heuritech Deep Learning Meetup #5," 29 February 2016. [Online]. Available: https://heuritech.wordpress.com/2016/02/29/a-brief-report-of-the-heuritech-deep-learning-meetup-5/. [Accessed 20 May 2023].

[37] B. Heo, S. Yun, D. Han, S. Chun, J. Choe and S. J. Oh, "Rethinking spatial dimensions of vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[38] T. George Karimpanal and R. Bouffanais, "Self-organizing maps for storage and transfer of knowledge in reinforcement learning," *Adaptive Behavior,* vol. 27, p. 111–126, 2019.

[39] J. Brownlee, "A Gentle Introduction to Transfer Learning for Deep Learning," 20 December 2017. [Online]. Available: https://machinelearningmastery.com/transfer-learning-for-deep-learning/. [Accessed 20 May 2023].

[40] Y. Sasaki and others, "The truth of the f-measure. 2007," *URL: https://www. cs. odu. edu/mukka/cs795sum09dm/Lecturenotes/Day3/F-measure-YS-26Oct07. pdf [accessed 2021-05-26],* vol. 49, 2007.

[41] "Classification: ROC curve and AUC - Google for Developers," [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc. [Accessed 20 05 2023].

[42] A. Jain, H. Patel, L. Nagalapatti, N. Gupta, S. Mehta, S. Guttula, S. Mujumdar, S. Afzal, R. Sharma Mittal and V. Munigala, "Overview and importance of data quality for machine learning tasks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

[43] A. Greco, A. Saggese, M. Vento and V. Vigilante, "Effective training of convolutional neural networks for age estimation based on knowledge distillation," *Neural Computing and Applications,* p. 1–16, 2021.

[44] Q. Cao, L. Shen, W. Xie, O. M. Parkhi and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*, 2018.

[45] R. Rothe, R. Timofte and L. Van Gool, "Deep expectation of real and apparent age from a single image without facial landmarks," *International Journal of Computer Vision,* vol. 126, p. 144–157, 2018.

[46] R. Rothe, R. Timofte and L. Van Gool, "Dex: Deep expectation of apparent age from a single image," in *Proceedings of the IEEE international conference on computer vision workshops*, 2015.

[47] Z. Zhang, Y. Song and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[48] L. Lakshmanan, "How to train a ResNet image classifier from scratch on TPUs on AI Platform," 10 July 2018. [Online]. Available: https://cloud.google.com/blog/products/ai-machine-learning/how-to-train-a-resnet-image-classifier-from-scratch-on-tpus-on-cloud-ml-engine. [Accessed 20 May 2023].

[49] C. Coleman, D. Narayanan, D. Kang, T. Zhao, J. Zhang, L. Nardi, P. Bailis, K. Olukotun, C. Ré and M. Zaharia, "Dawnbench: An end-to-end deep learning benchmark and competition," *Training,* vol. 100, p. 102, 2017.

[50] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV),* vol. 115, pp. 211-252, 2015.

[51] "Kneron's Edge AI & facial recognition survey pushes forward industry," 16 January 2020. [Online]. Available: https://www.kneron.com/read/85/. [Accessed 20 May 2023].

[52] Daniel Simpson, Meghana Athavale, Alex Buck, J.P. DeLauri, Meera Dietzel, Sarah Vilaysom, Terry Lanfear, "Threat modeling AI/ML systems and dependencies," 2022.

[53] A. Ilyas, L. Engstrom, A. Athalye and J. Lin, "Black-box adversarial attacks with limited queries and information," in *International conference on machine learning*, 2018.

[54] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *Ieee Access,* vol. 6, p. 14410–14430, 2018.

[55] S. Tariq, S. Jeon and S. S. Woo, "Am I a Real or Fake Celebrity? Measuring Commercial Face Recognition Web APIs under Deepfake Impersonation Attack," *arXiv preprint arXiv:2103.00847,* 2021.

[56] "Amazon SageMaker Pricing," [Online]. Available: https://aws.amazon.com/sagemaker/pricing/. [Accessed 20 May 2023].

[57] "Amazon EC2 On-Demand Pricing," [Online]. Available: https://aws.amazon.com/ec2/pricing/on-demand/. [Accessed 20 May 2023].

[58] "Models and pre-trained weights — Torchvision main documentation," 2017. [Online]. Available: https://pytorch.org/vision/main/models.html. [Accessed 20 May 2023].

[59] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013.

[60] "Sklearn.Model_selection.Train_test_split," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. [Accessed 20 May 2023].

[61] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580,* 2012.

[62] Z. Bao, Z. Tan, Y. Zhu, J. Wan, X. Ma, Z. Lei and G. Guo, "LAE: long-tailed age estimation," in *Computer Analysis of Images and Patterns: 19th International Conference, CAIP 2021, Virtual Event, September 28–30, 2021, Proceedings, Part II 19*, 2021.

[63] R. Stojnic, R. Taylor, M. Kardas, E. Saravia, G. Cucurull, Andrew and T. Scialom, "Papers with code - ImageNet (finetuned) benchmark (self-supervised image classification)," [Online]. Available: https://paperswithcode.com/sota/self-supervised-image-classification-on-1. [Accessed 20 May 2023].

RAW DATA OF RESNET-50 MODEL TRAINING AND VALIDATION LOSS APPENDIX 1

Epoch	Training loss	Validation loss
1	15.24029	76.61787
2	11.02026	70.96482
3	8.07648	70.98355
4	5.883182	74.10122
5	4.314426	73.15331
6	3.160482	73.11796
7	2.324925	70.0954
8	1.724124	70.8137
9	1.309423	70.02855
10	1.216326	70.92981
11	1.016791	70.8075
12	0.814965	72.0695
13	0.76869	72.13709
14	0.674138	72.56792
15	0.568866	73.10345
16	0.543387	72.17797
17	0.493004	71.10126
18	0.43754	71.93923
19	0.422517	71.70122
20	0.397911	71.27272
21	0.361613	72.49165
22	0.351609	72.02167
23	0.33377	72.49143
24	0.313143	71.83089
25	0.306548	72.19706
26	0.297665	71.89792
27	0.281616	72.56798
28	0.275825	72.14204
29	0.267533	71.73945
30	0.256485	72.36282
31	0.251516	72.09731
32	0.245895	72.08474

33	0.236149	72.20223
34	0.232593	71.87704
35	0.228198	71.02522
36	0.220131	72.19598
37	0.217766	71.61704
38	0.213584	72.04855
39	0.209151	71.1984
40	0.206234	71.95555
41	0.203064	71.83209
42	0.199135	72.74314
43	0.197067	71.94033
44	0.195295	71.0783
45	0.190775	72.10779
46	0.188238	71.93627
47	0.184684	72.12284
48	0.182199	72.43616
49	0.178713	72.69801
50	0.175476	72.15043
51	0.171702	72.36977
52	0.1665	71.93978