721 Project 1: Facial Recogition

Teresa Zhang   
 The University of Auckland  
COMPSY DepartementAuckland, New Zealand

*Abstract*— Facial recognition is a widely adopted biometric technology that identifies or verifies a person’s identity based on their unique facial features. CNN is the corn technology for facial recognition, which is designed to learn the feature from visual data. To contribute to the development of the effective and robust CNN model for facial recognition, this report presents a comprehensive study focused on the model architecture and dataset evaluation. This report includes the literature review of the three commonly used CNN model: VGG, ResNet and Inception-v4, and the four public facial datasets: LFW, VGGFace2, Megaface and CASIA-WbeFace. Following the literature analysis, two representative models, VGG9 and ResNet18, are selected for implementation. These models are trained and fine-tuned on the Labeled Faces in the Wild (LFW) dataset. A performance evaluation and analysis based on key metrics is provided in the report to assess the effectiveness of each model under practical conditions.

Keywords—CNN, facial recognition, LFW dataset

# Introduction

With the rapid advancement of technology, the risk of online personal private information leaking has become an important issue. In response, facial recognition has emerged as a key method for identity verification and security screening. Facial recognition is convenience, speed, and secure technique, which make it one of the most widely adopted forms of identity verification method today. The development of modern facial recognition systems relies heavily on machine learning, particularly Convolutional Neural Networks (CNNs), which have become the core architecture for extracting and learning discriminative facial features from visual data.

A table of data with numbers and letters

AI-generated content may be incorrect.

CNN is a type of deep learning model specially designed to process and analyze visual data, such as images or videos. The CNN usually consists of four types of layers: convolutional layers, pooling layers, fully connected layers and SoftMax layer. The convolutional layers is the key to the CNN. Each convolutional layers detect the features by applying filters (kernels) to the image. By stacking multiple convolutional layers, the CNN model able to learn feature and patterns automatically from raw pixels.

Since the first CNN was introduced in 1990s, there are more and more CNN architecture been developed. Therefore, conducting a comparative analysis of commonly used CNN models, evaluating their strengths and weaknesses is crucial for advancing facial recognition research and applications. In this report, I present a literature review for the architecture of three representative deep learning models—Inception-v4, ResNet, and VGG in part 2. In part 3, I compared and discussed four widely used, publicly available human face dataset. The goal is to assess their performance, complexity, and suitability for different facial recognition scenarios. In part 4, I train two selected CNN models on LFW dataset, with a clearly described methodology. I analysis the performance of models in part 5.

# modle comparision

## VGG

VGG is a standard deep CNN model with multiple convolutional layers. It use a very small (3 × 3) convolution filters through entire neural network, which decreases the complexity of model, reduce the trainable parameters, makes it easy to understand and implement.[3] It provides a strong baseline performance. Now a days, it is still effective on many tasks despite newer architecture being more advanced.

The architecture of VGG is very straightforward: It usually consists of multiple CNN layers stacked deeper and deeper, with pooling layers (2×2 max pooling) to reduce image size and prevent overfitting. The network concludes with fully connected layers, followed by a SoftMax classifier for final prediction. There are usually 3 layers for fully connected layer with SoftMax, therefore the depth of the VGG is dependent on the convolutional layer. E.g. VGG 16 has 13 convolutional layers and 3 fully connected layers. [2] The architecture for different types of VGG is present in figure 1 below.

1. FConvNet configurations for VGG [3]

## ResNet [4]

When the technology of neural networks improving, the model tends to build deeper and deeper. However, the accuracy starts to saturated due to the vanishing gradient with the growth of model depth. ResNet introduced a new concept of residual blocks to address this problem. In the residual blocks, a shortcut technique is used to skip the connections for one or more layers as shown in figure 2 below. The residual blocks enable the networks to learn identity mappings, ensuring that deeper layers do not degrade performance, and allows them to build deep networks with hundreds of layers. The deep ResNet can benefit from an increase of depth without worrying about vanishing gradient.

A diagram of a diagram

AI-generated content may be incorrect.

1. Residual learning: a building block[4]

## Inception-v4[1]

The Inception network is composed by multiple inceptions block, which divide the input into parallel convolution path and subsequently merges the output. This multi-branch structure enables the network to capture different features at spatial scales. Compare to the early version (inceptionv1 to v3)\_, the inception v4 doesn’t incorporate residual connection. Instead, it improved the accuracy by shedding unnecessary baggage and standardizing the design. Specifically, Inception-v4 use a uniform grid size across inception blocks. enhancing structural consistency and optimization.

The overall architecture consists of a stem module, followed by repeated Inception-A, Reduction-A, Inception-B, Reduction-B, and Inception-C blocks. This modular and well-balanced design allows Inception-v4 to achieve high performance on large-scale image recognition tasks, while maintaining computational efficiency and architectural clarity.

## Comparison

Based on the literature review above, VGG, ResNet and Inception-v4 represent three influential CNN architectures with different design philosophies. Compare to other CNN model, VGG has a comparable straightforward architecture that consists of deep stacks of 3 × 3 convolutions layer. Its architecture provides a strong baseline performance but higher computational complexity due to the larger number of trainable parameter. In other hand, the ResNet introduce the residual connection that enabling deeper networks while significantly reducing model size and improving training efficiency. Inception-v4 enhances recognition accuracy by using multi-branch that process features at multiple scales. Its architecture allows to learn more complex pattern but has higher computationally demanding.

The computational complexity and parameter count of models are summarized in Table 1. All models were trained and evaluated on the ImageNet-1k dataset using an NVIDIA Titan X Pascal GPU. As shown, VGG exhibits the highest number of parameters and FLOPs, reflecting its large model size and computational cost. Inception-v4 has a similar parameter count comparable to ResNet, but with significantly higher FLOPs, indicating greater computational complexity despite similar model size. Overall, ResNet achieves a balanced trade-off with moderate model size and the lowest computational demand among the three.[5]

1. Computational Complexity Comparison [5]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | | Layers | Params(M) | FLOPs(G) | Memory Usage (GB) |
| VGG | VGG13 | 19 | 133 | 19.63 | 1.47 |
| VGG16 | 16 | 138 | 31.00 | 1.51 |
| VGG19 | 19 | 144 | 39.00 | 1.52 |
| ResNet | ResNet-34 | 34 | 21.8 | 3.68 | 0.91 |
| ResNet-50 | 50 | 25.6 | 4.09 | 0.95 |
| ResNet-101 | 101 | 44.5 | 7.58 | 1.08 |
| Inception-V4 | | ~75 | 42.7 | 13.00 | 1.58 |

The performance is also a critical parts for model qualitative comparison. However, the performance of the model is highly related to the dataset they been trained. There are four study that trained and evaluated the result of model in different dataset is shown in table 2 below.

On large-scale datasets such as ImageNet-1k, Inception-v4 consistently outperforms both VGG and ResNet due to its multi-scale feature extraction and architectural depth.[5] However, in simpler or smaller datasets like LFW or CIFAR-100, ResNet and VGG show more stable and efficient performance.

The study from Gwyn, Tony, et al [6] showed that the VGG 16 performs best on LFW in face recognition tasks. ResNet-101 achieves the best accuracy on CIFAR-100 dataset [7] and ResNet-50 excels in domain-specific datasets like rice disease classification [8]. Given the relatively small size of the LFW dataset, higher-complexity models such as Inception-v4 are not well-suited for this task. Therefore, VGG and ResNet were selected for training due to their moderate depth, lower complexity, and proven effectiveness on smaller datasets.

1. Performance Comparison

|  |  |  |  |
| --- | --- | --- | --- |
| Paper | Dataset | Models compared | Top Performer |
| Benchmark Analysis.[5] | ImageNet-1k | VGG,ResNet, Inception-v4 | Inception-v4 |
| Face Recognition [6] | LFW | VGG-16/19, ResNet50/101, Inception v2/ v3 | VGG-16 |
| Comparative Analysis [7] | CIFAR-100 | VGG,ResNet-101, Inception-v4 | ResNet-101 |
| Comparison of rice diseases [8] | Rice Leaf Disease | VGG16/19, InceptionV3, ResNet50 | ResNet50 |

In this study only the top 8 classes from the LFW dataset are considered, which further reduces the dataset size and complexity. While high-capacity models typically achieve superior performance on large and diverse datasets, they often underperform on small or simplified datasets due to overfitting. In such cases, lightweight architectures like VGG 11, with fewer layers and parameters, offer better generalization. Its reduced complexity enables more efficient learning from limited data, whereas deeper models such as VGG16 may capture noise rather than meaningful features, leading to degraded performance [12].

Therefore, a custom VGG 9 model and ResNet18 were selected for training and evaluation. The VGG9 model is a simplified, self-designed variant based on the VGG11 architecture, aimed at further reducing model complexity. It consists of a total of 9 layers, including 6 convolutional layers followed by 3 fully connected layers. This streamlined structure allows for faster training and reduces the risk of overfitting, particularly on smaller datasets. The detailed architecture of VGG9 will be presented in Part 4 of this report. In conclusion, the lightweight design of both VGG 9 and ResNet 18 provide an optimal balance between model capacity and generalization ability, making them well-suited for the simplified version of the LFW dataset used in this study.

# Dataset review

To support the development and evaluation of facial recognition technique, a variety of publicly available human face datasets have been established. This section presents a qualitative comparison of four widely used datasets: Labeled Faces in the Wild (LFW), VGGFace2, MegaFace, and CASIA-WebFace.

## LFW [6]

The Labelled Face in the Wild (LFW) is the one of the earliest and most widely used benchmarks dataset for evaluating facial recognition algorithms. There are 13,233 face image for 5,749 individuals, all size in 250 × 250. The image in LFW is captured celebrities in head shot, collected from the web. The background of the LFW is from the real-world environment, which includes natural variations in lighting, pose, background.

## VGGFace2 [9]

VGGFace2, developed by the Visual Geometry Group at the University of Oxford, is a large-scale dataset designed specifically for training robust face recognition models. It includes over 3.3 million images from 9,131 identities, with between 80 and 800 images for each identity. The dataset collection is proposed that encourages diversity in pose, age, ethnicity and facial expression for each subject. strongly improve the robustness of the dataset. Nowadays, VGGFace2 has become a popular benchmark in the research community.

## Megaface [10]

MegaFace is built from Yahoo’s 100M Flickr dataset and includes over 1 million face images that capture more than 690,000 unique individuals. MegaFace is unique in that it challenges algorithms with open-set identification tasks, where the gallery contains many identities, but only a small portion are known. Another unique point for MegaFace datasets is that they tend to be broad rather than deep. In other words, the dataset contains a high number of individuals but less image for everyone. While MegaFace is not typically used for training due to noisy labels, it is considered a gold standard for stress-testing recognition models at scale.

## CASIA-WebFace [11]

CASIA-WebFace is a medium-scale dataset curated by the Chinese Academy of Sciences, containing 494,414 images across 10,575 identities, mostly of celebrities. The dataset features in-the-wild images with moderate variation in pose, lighting, and background, offering a balance between scale and diversity. The annotation of the CASIA-WebFace is done by semi-automatic via clustering and name matching and manually correct afterward. Therefore, it will contain some noisy labels. CASIA-WebFace has been widely used as a pretraining dataset for CNN-based facial recognition models, especially prior to the availability of larger datasets like VGGFace2 or MS-Celeb-1M.

## Comparison

The above four unique human faces datasets have their own strengths and weaknesses. All the datasets are collected in real-world background with environmental noise effects, introduce similar robustness. Therefore, this comparison focuses more on the number of classes, image distribution and the resolution of image. The basic features of datasets are organized in table 3 below:

1. feature comparison for datast

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset | Dataset feature | | | |
| Classes | Images | Image per classes | Image size |
| LFW | 5,749 | 13,233 | 1~530 | ~250×250 |
| VGGFace2 | 9,131 | 3.3M | 80~800 | 224×224 |
| Megaface | ~690,000 | 1M | 50 | >100×100 |
| CASIA-WebFace | 10,575 | 494,414 | 1~804 | ~250×250 |

Based on the data and literature review above, LFW is best suited for benchmarking due to its real-world image conditions. The image in LFW dataset has a 250×250 high resolution, indicate the strong robustness of the dataset. But the low intra-identity variation, relatively small number of total classes and highly imbalance between classes somehow restricted the diversity.

On the other hand, VGGFace2 offers rich intra-person variability and balanced class distribution. The high-resolution images standardized at 224×224 pixels. In a word, it provides an outstanding performance in the field of training and evaluating deep learning models.

MegaFace has the highest number of classes and the lowest image size in four datasets, which brings it strong diversity but low robustness. However, it is a standout for its ability to test model performance under large-scale identification settings with massive distractor sets, and it is less suitable for training.

Finally, CASIA-WebFace offers a compromise between scale and accessibility, with enough diversity for training while remaining manageable in size. It has a more uneven distribution between classes (1~804).

Overall, VGGFace2 offers the best balance between resolution, class consistency, and variability, while LFW and MegaFace serve best for benchmarking under constrained and large-scale settings, respectively. CASIA-WebFace stands as a middle-ground option, offering accessible scale and reasonable diversity for training baseline models. The MegaFace CASIA-WebFace dataset has higher diversity due to their higher total classes, offering broader coverage of identities. In other hand, LFW and VGGFace2 demonstrate a higher robustness as they contain higher image qualities with greater variation in pose, lighting, and expression. Since this study focuses on evaluating CNN models on a small-scale dataset, LFW is the most suitable choice. It provides consistent image quality, real-world variations, and a manageable dataset size that suitable for the scope of lightweight model evaluation.

# methodology

In this section, we detail the data preprocessing pipeline, the architecture and implementation of the two selected models: VGG9 and ResNet18, as well as the hyperparameter tuning process used to optimize their performance.

## Data preprocessing

### Top 8 classes selected

The top 8 classes with higher number of images from the LFW dataset are selected for training. The details of each class are shown in table 4 below:

|  |  |
| --- | --- |
| Classes | Sample Size |
| George\_W\_Bush | 530 |
| Colin\_Powell | 236 |
| Tony\_Blair | 144 |
| Donald\_Rumsfeld | 121 |
| Gerhard\_Schroeder | 109 |
| Ariel\_Sharon | 77 |
| Hugo\_Chavez | 71 |
| Junichiro\_Koizumi | 60 |

### Resize image

The raw image size is 250 × 250, to fits in the two model we selected, the image is resized to ensure the batch shapes are consistent. To extract more information from the image and enhance the performance of CNN models, the size of 224 × 224 is picked.

### Image normalisation

The image is converted to the Pytorch tensor with values in [0.0,1.0]. Normalization ensures the CNN models learn efficiently and effectively, preventing large input values from dominating learning, and speeds up convergence. In RGB image, some channels might bring bias toward. Normalization ensures each channel contributes equally.

### Modifiy the image

Since the smallest classes in the data only contain 60 images, there is a potential risk of overfitting. To increase the diversity and improve generalization, several data augmentation techniques are applied to the training images. These include random horizontal flipping, random rotation up to 10 degrees, and slight adjustments to brightness and contrast (±10%). These modifications encourage models to ignore the effects from environmental noise and learn the pattern of the image.

### Data splitting

I split 70% data for training, 15% data for testing and 15% data for validation. It provides enough data for feature extraction during training and balances the validation variation and evaluation. I generate a fixed number of random seeds for splitting to ensure the results of the training consistency. The dataset is stratified which ensures that all classes are evenly represented across train, validation, and test sets by their weight.

## Model implementation

### VGG 9

The VGG9 model is a custom convolutional neural network inspired by the original VGG architecture but uses 9 weight layers in total — consisting of 6 convolutional layers and 3 fully connected (FC) layers. It is structured into three main convolutional blocks followed by a classifier. The first blocks apply a 3×3 convolution filter on 3-channel input (RGB image) and produce 64 feature maps. The non-linearity is introduced by ReLU function. Another convolution layer with same number of feature maps (64) stacked to increase the abstraction. At the end of the block, a Max pooling layer in stride of 2 applied to reduce the spatial dimension by half.

The second and third convolution blocks follow the same structure but double the size of the feature map every block. In block 3, there are 256 feature maps. After the convolutional feature maps are extracted, the model flattens them and passes through fully connected layers to perform classification. The first and second FC layers have 512 units, followed by ReLU and a default 0.5 dropout to prevent overfitting by randomly disabling neurons during training. The third FC layer outputs the classification for the input.

The model is designed to perform deep feature extraction using stacked 3×3 convolutions while maintaining a relatively lightweight architecture. With only 9 layers, it balances between expressive power and computational efficiency, making it well-suited for moderately sized datasets. The use of max pooling reduces spatial dimensions progressively, and dropout in the fully connected layers helps mitigate overfitting**.**

### ResNet18

As the name suggests, ResNet-18 consists of 18 layers. The model begins with a stem block that performs initial feature extraction and spatial down sampling. The stem block includes a 7×7 convolutional layer with stride 2 and padding 3 to capture broad spatial features from the input image. Following by batch normalization and ReLU activation that introduce non-linearity and stabilize training. End with a 3×3 Max Pooling layer with stride 2 further reduces the spatial resolution, preparing the feature map for deeper processing.

The core of the model consists of four residual stages, each built using the Basic Block structure, which includes skip connections that add the input to the output of the block to form residual connections:

* **Layer 1**: Two residual blocks with 64 filters (stride=1).
* **Layer 2**: Two residual blocks with 128 filters (stride=2).
* **Layer 3**: Two residual blocks with 256 filters.
* **Layer 4**: Two residual blocks with 512 filters.

The filter number is double after every layer to capture more feature. These residual connections help preserve gradient flow and make it easier for the network to learn identity mappings, improving training stability and reduce the computation complexity.

After the residual layers, a global average pooling layer is used to reduces each feature map to a single value, yielding a compact 512-dimensional representation per input. A dropout layer with the specified rate (default 0.5) is used to reduce overfitting. Finally, a fully connected layer maps the pooled features to the target number of classes. In this case the default number of classes is 8.

The model is design to benefit from the deep feature extraction while maintaining efficient gradient propagation through residual shortcuts. It balances depth and complexity with only 18 layers, making it suitable for moderately sized datasets. The use of global average pooling instead of flattening avoids large fully connected layers and reduces overfitting risk.

## Hyperparameter tunning

Hyperparameter tuning plays a critical role in optimizing model performance by finding the most suitable settings for training. In this project, tuning was focused on four key hyperparameters: batch size, learning rate, dropout rate, and number of epochs. These parameters significantly influence both the training dynamics and the generalization ability of the model.

The batch size controls how many samples the model processes before updating its weights. A batch size of 32 was found to work best for both VGG9 and ResNet18. This value strikes a good balance between gradient stability and training speed. Smaller batch sizes, such as 16, tended to introduce more noise into the gradient updates, which slowed convergence. On the other hand, larger batch sizes like 64 caused overfitting due to smoother but less frequent updates.

The learning rate controls the step size during optimization. A value that is too high may cause the model to overshoot minima, while one that is too low can result in slow convergence or getting stuck in local minima. Through tuning, a learning rate of 0.001 consistently yielded the best results across both models, avoiding instability while allowing the models to make meaningful parameter updates each iteration.

For the dropout rate, which is a regularization technique to prevent overfitting, the optimal values differed between the two models. The VGG9 model performed best with a dropout rate of 0.3, while ResNet18 achieved better results with a dropout rate of 0.5. This difference can be attributed to their architecture. VGG9 has fewer parameters than ResNet18 and tends to overfit less, so a lower dropout rate preserves more capacity. On the other hand, ResNet18, being deeper and more expressive, benefits from a higher dropout rate to reduce overfitting and improve generalization.

Lastly, the number of training epochs determines how many full passes the model makes over the training data. VGG9 reached optimal performance at 40 epochs, while ResNet18 performed best at 30 epochs. This could cause by the structure of VGG 9, which lack in shortcut connections and learns features more slowly. Therefore, VGG9 needs more epochs to fully converge. ResNet18’s residual connections improve gradient flow and accelerate learning, allowing it to converge earlier without sacrificing accuracy.

# Performance Comparison

In this section, we compare the performance of VGG9 and ResNet18 based on their accuracy. The evaluation metrics include precision, recall, and F1-score, derived from the models' performance on the test dataset, which is reserved exclusively for final performance evaluation. Additionally, we examine the training and validation accuracy and loss curves for both models to assess their learning behaviour and generalization capability.

### A. Testing dataset

For better understanding, the concept of the evaluation metrics introduced first.:

**Accuracy**: Measures the percentage of correctly predicted samples out of the total number of predictions.

**Precision**: Indicates how many of the predicted positive instances are correct. It is useful when the cost of a false positive is high.

**Recall**: Represents how many of the actual positive cases the model was able to identify. This is important when missing a positive case is costly.

**F1-Score**: The harmonic mean of precision and recall, providing a balance between the two.

The evaluation metric result for both model is show in the table 5 below. There are 203 samples in total is used for evaluation, which is the 15% of the total samples.

1. Evaluation Metrics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Classes | precision | recall | f1-score | Model | support |
| Ariel Sharon | 1.0000 | 0.9091 | 0.9524 | VGG9 | 11 |
| 1.0000 | 0.9091 | 0.8414 | ResNet18 |
| Colin Powel1 | 0.9730 | 1.0000 | 0.9863 | VGG9 | 36 |
| 0.8714 | 0.9444 | 0.9577 | ResNet18 |
| Donald Rumsfeld | 1.0000 | 0.9474 | 0.9730 | VGG9 | 19 |
| 0.9500 | 1.0000 | 0.9744 | ResNet18 |
| George W Bush | 0.9753 | 0.9875 | 0.9814 | VGG9 | 80 |
| 0.9753 | 0.9875 | 0.9814 | ResNet18 |
| Gerhard Schroeder | 1.0000 | 1.0000 | 1.0000 | VGG9 | 16 |
| 0.9375 | 0.9375 | 0.9375 | ResNet18 |
| Hugo Chavez | 1.0000 | 1.0000 | 1.0000 | VGG9 | 10 |
| 1.0000 | 0.8000 | 0.8889 | ResNet18 |
| Junichiro Koizumi | 1.0000 | 1.0000 | 1.0000 | VGG9 | 9 |
| 1.0000 | 1.0000 | 1.0000 | ResNet18 |
| Tony Blair | 1.0000 | 0.9545 | 0.9545 | VGG9 | 22 |
| 0.8750 | 0.9545 | 0.9130 | ResNet18 |
| macro avg | 0.9879 | 0.9748 | 0.9809 | VGG9 | 203 |
| 0.9637 | 0.9416 | 0.9507 | ResNet18 |
| weighted avg | 0.9806 | 0.9803 | 0.9802 | VGG9 | 203 |
| 0.9620 | 0.9606 | 0.9604 | ResNet18 |

Both models perform well across all classes, showing that the architectures are capable of learning meaningful features from the dataset. It clearly shows that the VGG 9 maintained consistently high precision, recall and F1-score nearly all classes, outperforming ResNet 18 in both macro-average and weighted-average metrics. The macro-average evaluates each class independently, while the weighted-average accounts for class imbalances by weighting metrics based on the number of true instances per class. Specifically, VGG 9’s weighted-average evaluation metrics were approximately 0.02 higher than those of ResNet18. This suggests that VGG 9 exhibits stronger generalization and stability when applied to unseen test data across all classes.

On other hand, ResNet 18 showed comparatively lower recall and F1 scores. This was especially noticeable for some classes such as Tony Blair and Hugo Chavez, which both have a relatively small amount of support samples. This indicates that ResNet 18 may be more sensitive to class imbalance or limited training data for certain categories, leading to reduced performance in those specific classes. It suggests that while ResNet 18 performs well overall, its performance might degrade for underrepresented classes.

Table 6 below summarizes the evaluation metrics for both models across the entire test dataset. While both models performed strongly, VGG 9 evaluation metrics were approximately 0.03 higher than those of ResNet 18. This clearly shows VGG 9 appears more suitable for this dataset and task based on its superior generalization performance.

|  |  |  |
| --- | --- | --- |
| Evaluation metric | Model | |
| VGG9 | ResNet18 |
| Precision | 0.9803 | 0.9606 |
| Recall | 0.9879 | 0.9637 |
| F1-score | 0.9748 | 0.9416 |
| Accuracy | 0.9809 | 0.9507 |

### B. Training and validating dataset

The figure 3 shows the accuracy and loss for VGG 9 through training and validation. The training accuracy steadily increases and converges around epoch 40. Meanwhile, the validation accuracy rises smoothly from 0.4 to around 0.8 and starts to fluctuate in a small variation around 0.8 to 0.9, indicating stable but limited generalization. The training loss decreases consistently to approximately 0.25, while the validation loss levels off between 0.25 and 0.50, showing a noticeable gap between training and validation loss. This persistent gap suggests that the model begins to overfit the training data despite overall strong performance.

A graph of a graph

AI-generated content may be incorrect.

1. VGG 9 train and validation accuracy and loss

The figure 4 below shows the accuracy and loss for ResNet 18 through training and validation.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

AI-generated content may be incorrect.

1. ResNet 18 train and validation accuracy and loss

Compared to VGG 9. ResNet 18 converges faster and initially superior training and validation accuracy. Both metrics approach near-perfect performance (around 0.975). However, validation accuracy exhibits significant fluctuations throughout the training process, and the validation loss varies sharply between 0.25 and 0.75. This wide range implies that ResNet 18 struggles with generalization and stability, likely due to overfitting.

Overall, the ResNet 18 exhibits a faster convergence speed and higher accuracy in training dataset. However, ResNet 18 shows significant instability in its validation accuracy and loss curve compared to VGG 9. In contrast, VGG 9 demonstrates more gradual and consistent improvement across both training and validation accuracy. This indicates that ResNet 18 updates its weights sharply and unstably. The significant variation in validation accuracy and loss in ResNet 18 suggests two potential issues: overfitting, where the model memorizes the noise instead of learning generalizable patterns; or optimization difficulties, likely due to the model’s deeper and more complex architecture, which may make it more sensitive to training parameters and regularization.

Interestingly, although ResNet 18 achieves a high validation accuracy of 97% during training, it underperforms on the test set compared to VGG 9. This further indicates the problem of overfitting. ResNet18 may have learned the validation data too well but failed to generalize to truly unseen samples.

Overall, VGG 9 proved to be the more reliable model in this experimental setup, balancing training stability and generalization effectively, while ResNet 18, although powerful, required more careful regularization and tuning to avoid overfitting.

# Conclusion

This report explored the foundations and developments in facial recognition through Convolutional Neural Networks (CNNs), focusing on their architectural design and effectiveness. By reviewing various CNN architecture including Inception-v4, ResNet, and VGG, and evaluating multiple human face datasets, the VGG 9 and ResNet 18 are selected to train and test on LFW dataset.

The experimental results showed that ResNet 18 achieved higher training and validation accuracy and demonstrated faster convergence. However, it struggled with generalization, as shown by the significant fluctuations in validation accuracy and loss, along with comparatively lower performance on the testing dataset. In contrast, VGG 9 exhibited more stable and consistent trends throughout validation, and testing. It outperformed ResNet-18 across all key evaluation metrics, including precision, recall, F1-score, and final test accuracy—indicating better real-world applicability.

These results can be largely attributed to the architectural differences between the two models. VGG 9 has a relatively simple and deep sequential structure. Its low architectural complexity might avoid to overfitting, allowing it to generalize better. The slower convergence speed due to the higher number of trainable parameters for VGG 9. On the other hand, ResNet 18’s residual blocks allow for faster convergence and deeper feature learning, but the added complexity can make the model more sensitive to noise and prone to overfitting, especially with a limited dataset like LFW.

In summary, VGG 9 is more suitable for facial recognition tasks that require consistent and reliable performance. ResNet-18 may be better suited for larger datasets or scenarios where rapid training is critical.

##### References

1. Szegedy, Christian, et al. “Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning.” *ArXiv:1602.07261 [Cs]*, 23 Aug. 2016, arxiv.org/abs/1602.07261v2.
2. Tammina, Srikanth. “Transfer Learning Using VGG-16 with Deep Convolutional Neural Network for Classifying Images.” *International Journal of Scientific and Research Publications (IJSRP)*, vol. 9, no. 10, 6 Oct. 2019, p. p9420, https://doi.org/10.29322/ijsrp.9.10.2019.p9420.
3. Simonyan, Karen, and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition.” *ArXiv.org*, 2014, arxiv.org/abs/1409.1556v6.
4. He, Kaiming, et al. “Deep Residual Learning for Image Recognition.” *Openaccess.thecvf.com*, 2016, openaccess.thecvf.com/content\_cvpr\_2016/html/He\_Deep\_Residual\_Learning\_CVPR\_2016\_paper.html.
5. Bianco, Simone, et al. “Benchmark Analysis of Representative Deep Neural Network Architectures.” *IEEE Access*, vol. 6, 2018, pp. 64270–64277, https://doi.org/10.1109/access.2018.2877890.
6. Gwyn, Tony, et al. “Face Recognition Using Popular Deep Net Architectures: A Brief Comparative Study.” *Future Internet*, vol. 13, no. 7, 25 June 2021, p. 164, www.mdpi.com/1999-5903/13/7/164, https://doi.org/10.3390/fi13070164. Accessed 30 Nov. 2021. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
7. Zhang, Xiao, et al. “Comparative Analysis of VGG, ResNet, and GoogLeNet Architectures Evaluating Performance, Computational Efficiency, and Convergence Rates.” Applied and Computational Engineering, vol. 44, no. 1, 5 Mar. 2024, pp. 172–181, www.ewadirect.com/proceedings/ace/article/view/10625, https://doi.org/10.54254/2755-2721/44/20230676.
8. Shah, Syed Rehan, et al. “Comparing Inception V3, VGG 16, VGG 19, CNN, and ResNet 50: A Case Study on Early Detection of a Rice Disease.” Agronomy, vol. 13, no. 6, 1 June 2023, p. 1633, www.mdpi.com/2073-4395/13/6/1633, https://doi.org/10.3390/agronomy13061633.
9. Cao, Qiong, et al. “VGGFace2: A Dataset for Recognising Faces across Pose and Age.” *ArXiv.org*, 2017, arxiv.org/abs/1710.08092?utm\_source=chatgpt.com. Accessed 5 May 2025.
10. Kemelmacher-Shlizerman, Ira, et al. “The MegaFace Benchmark: 1 Million Faces for Recognition at Scale.” *Openaccess.thecvf.com*, 2016, openaccess.thecvf.com/content\_cvpr\_2016/html/Kemelmacher-Shlizerman\_The\_MegaFace\_Benchmark\_CVPR\_2016\_paper.html.
11. Yi, Dong, et al. “Learning Face Representation from Scratch.” *ArXiv.org*, 2025, arxiv.org/abs/1411.7923?utm\_source=chatgpt.com.
12. Zhao, Peng, et al. “A Comparative Study of Deep Learning Classification Methods on a Small Environmental Microorganism Image Dataset (EMDS-6): From Convolutional Neural Networks to Visual Transformers.” *Frontiers in Microbiology*, vol. 13, 2 Mar. 2022, pmc.ncbi.nlm.nih.gov/articles/PMC8924496/?utm\_source=chatgpt.com, https://doi.org/10.3389/fmicb.2022.792166. Accessed 25 Apr. 2022.