Doctor of Philosophy

High Performance Face Identification System with Optimized Deep Learning Architectures

The Graduate School

of the University of Ulsan

Department of Electrical/Electronic and Computer Engineering

Laksono Kurnianggoro

High Performance Face Identification System with Optimized Deep Learning Architectures

Supervisor: Kang-Hyun Jo

A Dissertation

Submitted to
the Graduate School of the University of Ulsan
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

by

Laksono Kurnianggoro

Department of Electrical/Electronic and Computer Engineering
University of Ulsan

December 2019

High Performance Face Identification System

with Optimized Deep Learning Architectures

This certifies that the dissertation

of Laksono Kurnianggoro is approved:

Committee Chair Prof. Hee-Jun Kang

Committee Member and Supervisor Prof. Kang-Hyun Jo

Committee Member Prof. Young-Soo Suh

Committee Member Prof. Hyun-Deok Kang

Committee Member Prof. Jang-Sik Park

Department of Electrical/Electronic and Computer Engineering

University of Ulsan, South Korea

December, 2019

"Those who can imagine anything, can create the impossible."

Alan Turing

UNIVERSITY OF ULSAN

ABSTRACT

Graduate School of Electrical Engineering

Department of Electrical/Electronic and Computer Engineering

Doctor of Philosophy

High Performance Face Identification System with Optimized Deep Learning Architectures

by Laksono Kurnianggoro

Human identification is an essential element in authentication systems and surveillance devices. Various technologies are utilizing this method to perform higher-level processing and provides additional values in their solutions such as in smartphone, automatic gate, attendance recorder, social media, and smart CCTV system.

The work on this manuscript focus on the camera-based identification system in a real-world scenario where human faces are the main feature. This kind of system requires a fast processing time as well as acceptable identification quality. To achieve these two requirements, the research in this study proposes efficient deep learning models for detecting and identifying faces from a given video stream.

A complete face identification system consists of face detection and face identification system. For the face detection module, the proposed model uses an altered version of the single-shot detection model. This model is optimized to detect not only large faces but also the small ones. By utilizing the low-level and high-level features, the model can deliver satisfying performance. Furthermore, an attention-based mechanism is incorporated to helps the training process of the model.

As the detection module predicts un-aligned faces, the identification result is compromised. Therefore, an alignment procedure is needed to rectify the predicted faces. Instead of using a separated model, this module is embedded directly in the face detection module. This kind of strategy allows the system works efficiently. Compared to the two-stages approach, where the detection and alignment process uses separated models, this strategy works more efficiently.

The face identification module extracts an embedding vector from a given face image. This module utilizes a deep learning model that extract distinctive features for each identity. It is achieved by train the model using the angular margin strategy to ensure the feature vector that corresponds to an identity form a distinct cluster in the embedding space. The model is then augmented using the auto-encoder principle to fortify the feature transformation result.

Despite the satisfying performance, the deep learning model often suffers in processing time. A model needs more layers to achieve better results, which leads to more processing time. Therefore, this study utilizes a channel pruning strategy to alleviate this problem. As a result, significant improvement in the processing speed is achieved, allowing the whole pipeline of the face identification model to be executed efficiently for real-world applications.

Acknowledgements

I would like to express my gratitude for my supervisor, Professor Kang-Hyun Jo, who gave me an opportunity to study under his guidance, nurture, encourage, and give abundant support during my life as a graduate student at the University of Ulsan. I would also thank the members of the thesis committee Prof. Hee Jun Kang, Prof. Young Soo Suh, Prof. Hyun-Deok Kang, and Prof. Jangsik Park for their invaluable comments and suggestion to improve the quality of this thesis.

I am grateful to the member of the Intelligent Systems Laboratory for the meaningful discussion and lesson that enhance my knowledge and broaden my understanding in this research field, computer vision and machine learning. Many thanks for the Brain Korea Scholarship program that allow me to attend various conferences around the world and expand my horizon as a student and researcher. Special thanks to my wife and my daughter who always stays beside me during my study In Korea. Finally, my grateful praise is for God, the Lord, and Nourisher of the worlds.

Contents

Abstract iii

Acknowledgements v

1 Introduction 1

1.1 Motivation and Background 1

1.2 Problem Description and Objective 3

1.3 Contributions 5

1.4 Disposition 5

2 Literature Review 7

2.1 Object Detection 7

2.1.1 Classical Object Detector 8

2.1.2 Deep Learning-based Object Detector 9

2.1.3 State-of-the-Arts in Face Detector 12

2.2 Face Recognition Network 14

2.3 Inference Optimization for Deep Learning Model 16

2.3.1 Network pruning 16

2.3.2 Neural Architecture Search 17

3 High Performance Face Detection Network 19

3.1 High Performance Face Detection System with Attention-Guided Model 20

3.1.1 System Architecture 21
3.1.2 Soft Attention Mask and Data Augmentation 23
3.1.3 Experiments and Results 24
3.2 Efficient Face Detector 25
3.2.1 Pruning Strategy 29
3.2.2 Experiments 30

4 Efficient Face Recognition Network **33**
4.1 Fast Inference using Pruned Model 35
4.2 Experiments and Results 37

5 Real-time Face Identification System **45**

6 Conclusion **50**
6.1 Future Works 52

A Publications **53**
A.1 Journal 53
A.2 Conference 53

Bibliography **59**

List of Figures

1.1 Illustration of a face identification system. 1

1.2 Top-20 cities surveilled by CCTV system. 2

1.3 Major components in an end-to-end face identification system. 4

2.1 An object detector with the sliding window strategy. 8

2.2 Overview of the R-CNN method. 10

2.3 Overview of the Faster-RCNN method. 11

2.4 Overview of the SSD method. 12

2.5 Sub-fields of the face recognition research. 15

3.1 Overview of the proposed method. 20

3.2 Architecture of the face detector network. 22

3.3 Example of the data augmentation result. 23

3.4 Overview of the proposed method. A deep learning model is designed to predict heatmap of faces as well as their corresponding bounding boxes. 26

3.5 Examples of the successful detection results. 27

3.6 Illustration of a pruned network. Black dashed-lines corresponds to a dropped node in the next layer. 28

3.7 Channel-wise pruning strategy. Removed weights and feature channels are black-colored. 28

3.8 Illustration of the iterative pruning strategy. 30

3.9 Effect of pruning rate to the system performance. 32

4.1 Embedding features extracted using a network trained on MNIST dataset using different types of loss functions. Left: softmax-loss, right: center loss, top: un-normalized, bottom: normalized. Best viewed in display. 34

4.2 Illustration of the relation between embedding features and their corresponding class-weight. Best viewed in display. 35

4.3 Illustration of the face identification using embedding features. Best viewed in display. 36

4.4 Pipeline for building the face recognition network. 37

4.5 Similarity score calculation. 38

4.6 Example of positive pairs from the LFW and CA-LFW dataset. Redrawn from the original paper (Zheng, Deng, and Hu, 2017). 39

4.7 Examples of negative pairs from the LFW and CA-LFW dataset. Redrawn from the original paper (Zheng, Deng, and Hu, 2017). 40

4.8 Evolution of the channel size across different pruning rate. Best viewed in display. 40

4.9 Performance of the Pruned VGG16 model on various benchmarks. Best viewed in display. 42

4.10 Performance comparison between iterative pruning and random initialization. Best viewed in display. 42

4.11 Performance of the pruned ResNet50 model on various benchmarks. Best viewed in display. 43

5.1 Processing speed comparison accross different setups (GTX1060Ti). . . 46

5.2 Processing speed comparison accross different setups (GTX1080Ti). . . 47

5.3 Examples of the face identification result. 49

List of Tables

3.1 Mean of Average Precision (mAP) of the Proposed Model at IoU 0.5. . 24

3.2 Peformance comparsion of the pruned S3FD models. 32

4.1 Performance evaluation of at different level of pruning rate on LFW dataset. 41

4.2 Performance comparison against state-of-the-arts models 44

List of Abbreviations

AI	**A**rtificial **I**ntelligence
CASIA	**C**hinese **A**cademy of **S**ciences, **I**nstitute of **A**utomation
CCTV	**C**losed **C**ircuit **Tele**vision
CNN	**C**onvolution **N**eural **N**etwork
CUDA	**C**ompute **U**nified **D**evice **A**rchitecture
CPU	**C**entral **P**rocessing **U**nit
FEM	**F**eature **E**nhancement **M**odule
FLOP	**FL**loating-point **OP**eration
FPN	**F**eature **P**yramid **N**etwork
fps	**f**rame **p**er **s**econd
GPU	**G**raphic **P**rocessing **U**nit
HOG	**H**istogram of **O**riented **G**radients
IoU	**I**ntersection **O**ver **U**nion
kNN	**k**-**N**earest **N**eighbours
NCS	**N**eural **C**opute **S**tick
mAP	**m**ean **A**verage **P**recision
MNIST	**M**odified **N**ational **I**nstitute of **S**tandards and **T**echnology
SSD	**S**ingle **S**hot **D**etector
SVM	**S**upport **V**ector **M**achine
RGB	**R**ed **G**reen **B**lue
ROI	**R**egion **O**f **I**nterest
VGA	**V**ideo **G**raphic **A**rray
VGG	**V**isual **G**eometry **G**roup
WIDER	**W**eb **I**mage **D**ataset for **E**vent **R**ecognition

List of Symbols

$\tilde{c}$	predicted class scores
$\tilde{o}$	predicted offsets
t	pruning step
s	importance score of a weight tensor
t	pruning step
x	input feature
y	target label
w	weight vector
A	Attention map
F	feature map
L	loss value from the objective function
W	weight tensor
∂	partial derivative
ϵ	small constant to avoid division by zero
σ	standard deviation
η	learning rate
τ	pruning ratio

Dedicated to
my family

Chapter 1

Introduction

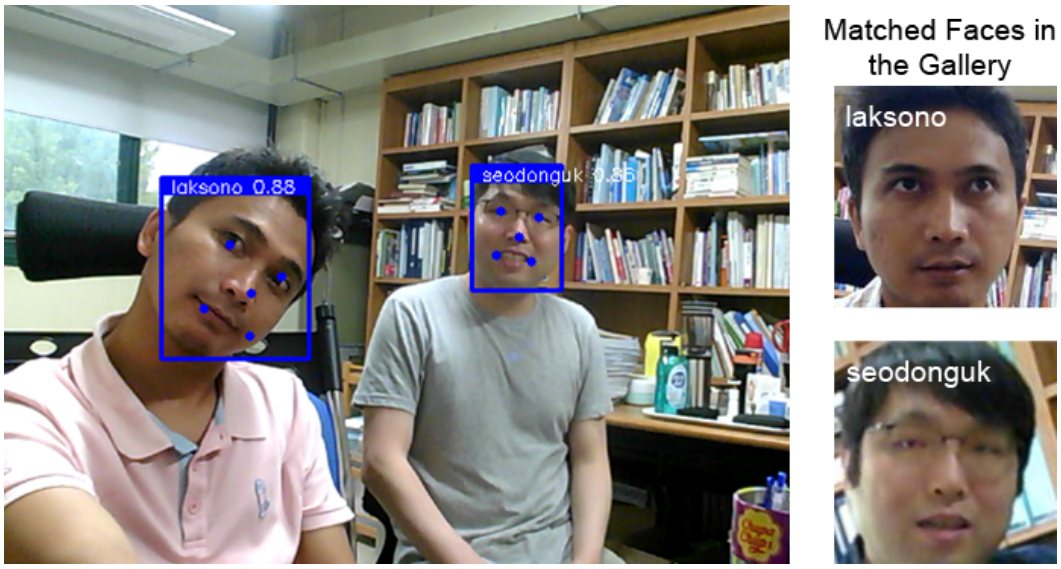

FIGURE 1.1: Illustration of a face identification system.

1.1 Motivation and Background

Human identification is a research field that grows continuously in the past few years. It becomes a useful technique that utilized in various sectors such as security, surveillance, military, and advertising. Several implementations are including unlocking a phone, finding a missing person, identification in forensic, attendance system, targeted advertising, and many more.

The face-based identification system is one of the approaches that getting a lot of attention in the past few years. It requires cheap sensors to operate and does not

need physical contact to work, unlike the fingerprint-based method. It works seamlessly to identify passerby, unlike the retina-based model that requires the people to maintain a specific pose.

The utilization of cameras in person identification has grown rapidly. According to a report published in the middle of 2019, China leads the world's CCTV surveillance system (Comparitech, 2019). Among the 120 cities worldwide, cities from China dominates in the top-10 ranking (Figure 1.2). Chongqing sits in the first ranking with approximately 2,579,890 cameras for more than 15,354,067 people. There are around 168 cameras in the city to monitor 1000 people. This number is huge compared to other modern cities such as Singapore (15.25 cameras/1000 people), Abu Dhabi (13.77 cameras/1000 people), Chicago (13.06 cameras/1000 people), and Moscow (11.70 cameras/1000 people).

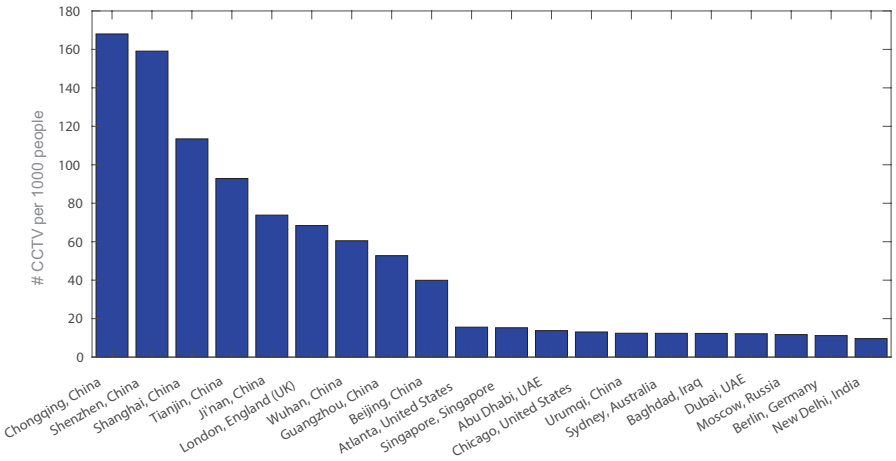

FIGURE 1.2: Top-20 cities surveilled by CCTV system.

The face identification system has been making new impacts in several cases. It helps the authority to catch a criminal on the run who had been evading justice for 20 years (SouthChinaMorningPost, 2019b). It also helps the police to identify jaywalkers and banned drivers (SouthChinaMorningPost, 2019c). Alongside the surveillance technology, companies started to use facial recognition in financial and advertisement. Merchants are using payment technology based on facial recognition instead of the barcode (SouthChinaMorningPost, 2019a). The payment process becomes seamless and more convenient for the buyers. In advertising, advertiser implements a new way to target ads at consumers. Using a camera installed nearby a display, it detects and identifies the viewer and shows personalized ads for them

based on gender, age, moods, and outside weather (TargetMarketing, 2019; BusinessNewsDaily, 2019).

Among the other computer vision methods, deep learning becomes the most popular technique for building face identification systems. It produces state-of-the-art results and outperforms other conventional methods by large margins, as reported in various research works (Parkhi, Vedaldi, and Zisserman, 2015, Zhang et al., 2017). In face detection, a model proposed by Zhang et al., 2017 achieves a 0.928 mAP score on the easy test set of WIDER face dataset (Yang et al., 2016). This score is far different compared to the score achieved by the well-known method proposed in Viola and Jones, 2001 (0.412 mAP score). Furthermore, a deep learning model achieves 97.3% accuracy on the face verification task (Parkhi, Vedaldi, and Zisserman, 2015), which is 13.5% larger compared to the Fisher Vector Face (Simonyan, Vedaldi, and Zisserman, 2014).

Albeit producing outstanding performances, deep learning models often suffer in processing time due to a large amount of computation burden. This slow computation restricts the applicability of deep learning models for the production-ready face identification system. Even though using a small deep learning model is possible, deeper models which demands more computation resources produce better results (He et al., 2016). Therefore, designing a model considering its trade-off between performance and computation cost is not a trivial task.

1.2 Problem Description and Objective

A complete face identification system consists of 2 main components, as illustrated in Figure 1.3. From an input image, a detection method is required to extract the region of interest (RoI) for each face. As the captured faces are not in a fixed pose, an alignment method is required to transform the faces into a canonical pose. Although this step is not absolutely necessary, (Parkhi, Vedaldi, and Zisserman, 2015) and (Schroff, Kalenichenko, and Philbin, 2015) show that this step increases the identification performance. Thus, this step is often incorporated alongside the face detector to build a robust face identification system. The last step is face identification which typically modeled as query matching between features from the extracted faces with features of faces stored in the database (gallery).

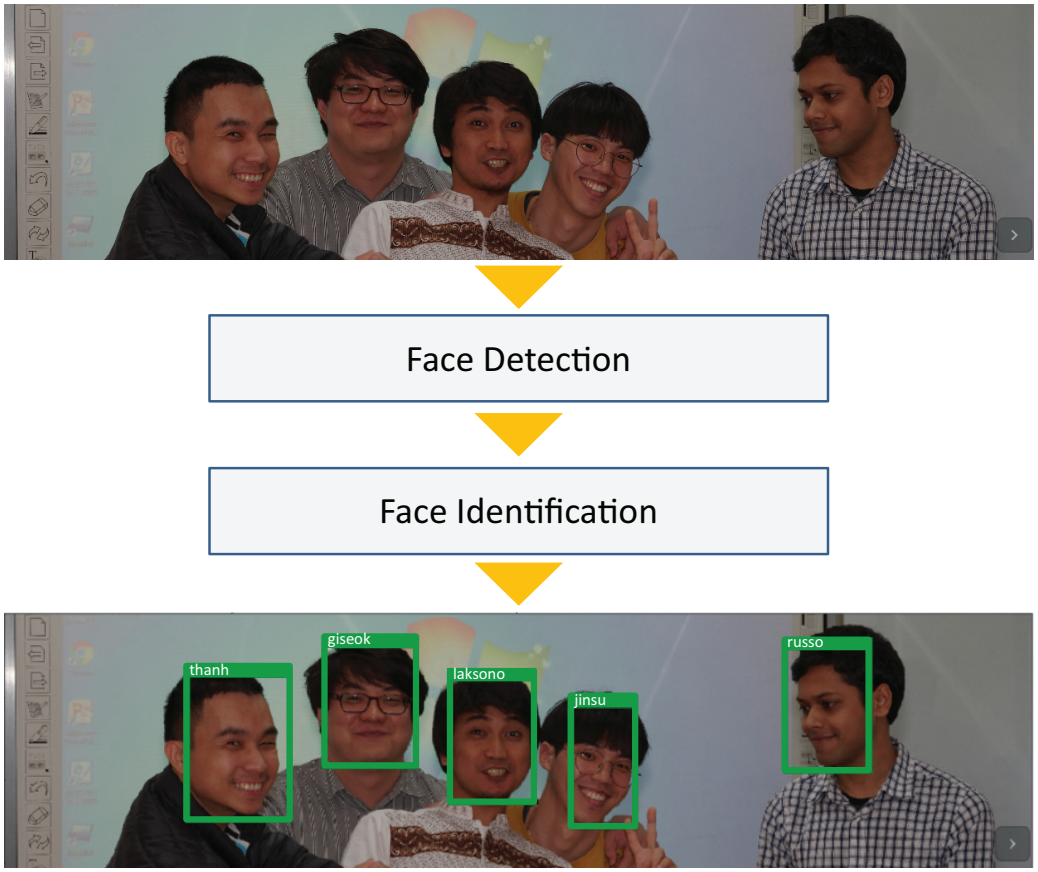

FIGURE 1.3: Major components in an end-to-end face identification system.

The work in this research aims to develop a reliable face identification pipeline, which is robust and runs within reasonable processing time. It uses deep learning models for all tasks to achieve high performance in terms of detection and identity verification. However, this means the system requires a large amount of computation cost which leads to slower processing time. To avoid this problem, the pruning method is incorporated to simplify the model architecture by removing some filters without sacrificing the system performance.

Performances of a deep learning model rely on various aspects including model architecture, dataset augmentation method, training strategy, and loss function. This works aims to improve the performance of detection and identification by exploring these aspects in several ways.

In face detection, a deep learning model often suffers in detecting small objects. The authors argue that high-level features that utilized for the final prediction losses important features from small objects since it encapsulates a large region in the input image. Therefore, this work expects a performance improvement by combining the

lower level features and high-level features for the final prediction step.

The second approach applied for face detection is utilizing the attention module to helps the training process. It helps the model to focus on important location in the image and ignore the less meaningful regions such as background parts.

1.3 Contributions

The work in this research explores various approaches in the development of a complete face identification pipeline. The contributions in this work are summarized as follows:

- Performance improvement in the face detector is achieved through the utilization of attention-based training and the usage of multiple scales enriched features.
- The predicted attention map makes the model more interpretable for observers.
- Extensive evaluations have been conducted in the widely used face detection benchmark and the result shows that the proposed method delivers comparable results against other states of the art methods.
- The proposed pruning procedure successfully reduces the processing time of the deep learning models. Thus, enabling a faster computation while maintaining system performance.
- This work demonstrates a complete pipeline of face identification method that works in real-time.

1.4 Disposition

This part explains the organization of this manuscript. The following section is discussing various methods related to the utilization of deep learning models in face detection and identification. It presents the classical detection model and the recent ones which are variant of the single-shot detection network. It also examines some

of the influential works on the face identification system. Furthermore, it is also presenting several methods related to model optimization, which are useful to enhance the computation speed.

Section 3 explains the proposed approach in developing a robust face detection model. It is including the architecture design, performance enhancement using the attention-based mechanism, channel pruning method, and integration of facial landmark detection that is useful for the alignment step. The last part of this section contains the discussion of performance evaluation on the proposed face detection model to reveal its advantages compared to other available methods.

Section 4 discusses the proposed deep learning method for face identification. This section contains several parts related to the proposed model, including system intuition, training objectives, and model pruning. Furthermore, it explores the performance assessment of the proposed method through various experiments.

Section 5 talks about the integration of the face detection and identification model as one complete pipeline. It is including the hardware selection, system setup, and discussion on the test results.

Finally, Section 6 presents a discussion of the possible future research direction and conclusion of this research work.

Chapter 2

Literature Review

The scope in this manuscript is considering three main components: face detection, identity retrieval, and inference optimization. This section discusses some of the publications in these topics that influence the recent research works.

This section consists of three parts. The first one covers a discussion about some methods affecting the current face detection. The second part presents several publications related to the face retrieval. Finally, the last one discusses some popular options to increase the inference speed of a deep learning model.

2.1 Object Detection

Object detection is a general method that is useful in analyzing a given input image. The research in this field has been growing in the past few years, with the help of machine learning. This approach generally consists of feature extraction and classification. Since the object appears in various sizes and positions, the object detector often employs a sliding window at multiple scale testing strategies.

Figure 2.1 illustrates the basic principle of an object detector. Firstly, the detector extracts some useful features from a region in the image. Then, a classifier determines whether that region contains an object or not based on its input features. The detector repeats this process from the top-left part until the bottom right part. As an object appears at an arbitrary scale, the sliding window process is repeated in several scales.

FIGURE 2.1: An object detector with the sliding window strategy.

2.1.1 Classical Object Detector

Feature engineering and machine learning model are crucial elements in the classical object detector. Feature engineering is a task-specific effort. Therefore, the designer needs to design and choose the features for a given task carefully. Furthermore, a decent amount of machine learning models were available. Hence, an object detector is often an output of mix-and-match between feature design and classifier selection.

One of the successful pipelines is the Viola&Jones face detector (Viola and Jones, 2001). It uses the haar-like feature and the AdaBoost classifier as its main components. This combination allows the proposed model runs in real-time, even in low-cost devices such as cameras.

The key to its fast processing time relies on the utilization of an integral image for fast computation. An integral image computes the total sum over a given region by accessing 4 locations in the summed area table.

Haar-like features compare the difference between values in a region compared to one or more regions nearby. Regions are compared based on pre-defined templates which are commonly a combination of rectangular shapes. The integral image computes efficiently a total value lies in a rectangular region. Therefore, the Viola&Jones face detector computes the required features efficiently by utilizing the integral image.

Classifications are performed on the Haar-like features using the threshold-based classifier. A single simple classifier is not robust to differentiate between face or non-face images. Therefore, Viola&Jones uses AdaBoost (Freund and Schapire, 1996; Freund, Schapire, and Abe, 1999) to combines the weak classifier into a strong final

classifier.

Other popular object detection pipelines are utilizing the Histogram of Oriented Gradient (HOG) feature (Dalal and Triggs, 2005) combined with a Support Vector Machine (SVM) classifier (Cortes and Vapnik, 1995). This method was designed to perform detection on a pedestrian dataset but can be adapted in other types of objects Wahyono, Hariyono, and Jo, 2017; Wahyono and Jo, 2017.

HOG calculates the gradient of an input image in the x-axis and y-axis. It then computes the magnitude of both gradients (the $L2$-norm) in each pixel as well as their corresponding orientation (i.e. $tan^{-1}g_x/g_y$ where g_i is the gradient in i-axis). This information is utilized to construct a histogram of magnitudes on each range of orientation.

One strong variant of HOG-SVM (Felzenszwalb et al., 2009) object detection is the Deformable Part Model (DPM). This method extends the Dalal-Triggs model (Dalal and Triggs, 2005) by detecting not only the main object but also several parts of the object. It then combines the root (main object) and its corresponding parts to form a stronger prediction. This model was tested on the Pascal VOC dataset which consists of 20 object categories.

A remarkable classical method in face detection is the Aggregate Channel Features (ACF) method (Yang et al., 2014). This method uses 10 types of features including LUV color channel, gradient magnitude, and 6-bins histogram from RGB color space. This method upgrades the weak decision stump classifier in Viola&Jones detector by a more complex 2-stages decision tree model. The AdaBoost method is then utilized to find the weights for each weak classifier. Finally, a soft cascade method (Bourdev and Brandt, 2005) is applied to produce the final prediction.

2.1.2 Deep Learning-based Object Detector

Deep learning gains a lot of attention in the past few years. Its main advantage is including automatic feature extraction, which eliminates the needs of manual feature engineering tasks. Another important factor in the rise of deep learning is the availability of GPU, which enables parallel computation at an affordable price. Thus, training a deeper model is getting easier and cheaper compared to two decades ago.

One variant of deep learning is the convolution neural network (CNN). The concept behind this model is dated back since the introduction of neocognitron (Fukushima, 1979) and one successful application in optical character recognition was published in the '90s (Lecun et al., 1998).

In 2012, a CNN model (Krizhevsky, Sutskever, and Hinton, 2012) wins the largest object recognition challenge, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). This success marks the starting point of the rise of the deep learning model in various computer vision tasks and machine learning research in general.

One of successful deep learning implementation for object detection is the Region with Convolutional Neural Network (R-CNN) model (Girshick et al., 2014). This model gathers a set of region proposals (i.e. potential ROIs) from the selective search algorithm (Uijlings et al., 2013). A CNN model then predicts the category of cropped and resized ROIs to determine what kind of object is shown inside. Additionally, it also predicts the bounding box refinement parameters for each cropped image.

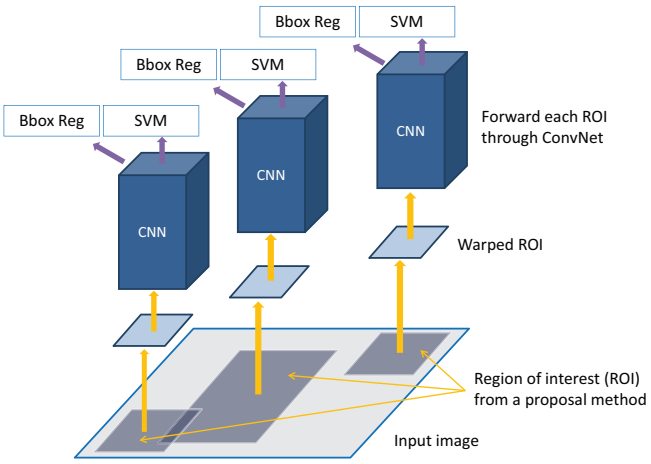

FIGURE 2.2: Overview of the R-CNN method.

Figure 2.2 illustrates the workflow of an R-CNN model. There are several problems with this scheme. Firstly, the ROIs are obtained from the external region proposal method. Secondly, the computation demanding CNN module extracts the features from each ROIs independently. Thus the amount of compassion is multiplied by the number of ROIs.

Several improvements were proposed to alleviate the limitations of the R-CNN model. Instead of cropping the ROIs and passing each crop to CNN independently,

the Fast-RCNN Girshick, 2015 proposes to perform cropping in the feature map generated by feeding the whole image into the CNN model. This allows a more efficient computation but the utilization of an external region proposal method is still becoming a bottleneck.

A region proposal network (RPN) is introduced in the Faster-RCNN model (Ren et al., 2015). This model augments the Fast-RCNN model by changing the external region proposal into an RPN module as illustrated in Figure 2.3. An RPN predicts the bounding box of potential regions from a given feature map. These regions are then cropped and passed into the ROI pooling module which warps all ROIs into the same size. The warped features are then passed into the final classifier which predicts the class and bounding box refinement parameters. As all components are differentiable, end-to-end training on this model is possible.

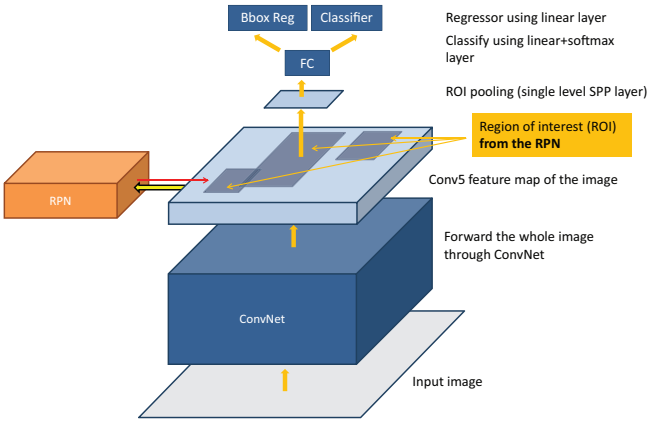

FIGURE 2.3: Overview of the Faster-RCNN method.

The single-stage pipelines (Liu et al., 2016; Redmon et al., 2016; Redmon and Farhadi, 2017) are introduced to perform fast inference in object detection. This type of method is predicting object classes inside the RPN module. Therefore, feature cropping and second stage detection are not performed. To covers the object prediction across various scales, prediction layers are attached to various levels of depth in the network.

Figure 2.4 illustrates an architecture of a single-shot detector (SSD) network. The network takes 300x300 pixels of an input image which is fed to the whole architecture. The original SSD architecture is consists of a VGG-16 backbone network, from the first layer until Conv5_3 layer, and several additional convolution layers. This model predicts the bounding box offsets and classes for each location in several feature maps including Conv4_1, conv7, conv8, conv9, conv10, and conv11.

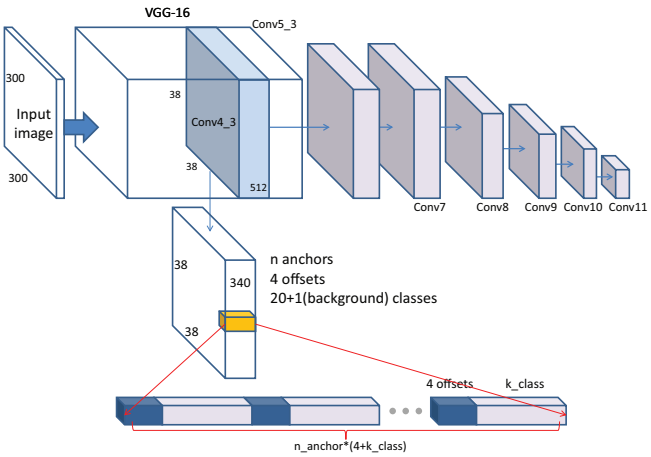

FIGURE 2.4: Overview of the SSD method.

In each prediction layer, the SSD model predicts 4 bounding box offsets and k+1 object classes (k objects, 1 background) for each anchor box. Anchor boxes are the predefined initial guess of the bounding box for each cell in the feature map. In the original experiments, SSD uses 4 or 6 anchor boxes for each prediction layer, resulting in a total of 8732 anchor boxes.

2.1.3 State-of-the-Arts in Face Detector

The single-shot scale-invariant face detector (S3FD) is an extension of the SSD model that specifically designed for face detection tasks (Zhang et al., 2017). There are several problems in training an SSD model for face detection task including detection of small-sized faces, huge amounts of negative anchors, and few amounts of anchor that matched the face. The S3FD method proposes several techniques to alleviate these problems.

S3FD method uses 6 prediction layers started in the Conv3_3 layer. This layer is 1 stride lower than the original SSD model which would detect faces from 160x160 cells. This detection layer is useful for predicting the small faces. Other detection layers are attached to the Conv4_3, Conv5_3, and 3 more layers at the extra convolution layers at stride 32, 64, and 128.

In the case of anchor design, S3FD uses anchor sizes that matched the size of the effective receptive field in each prediction layer. Specifically, it uses an anchor size of 16, 32, 64, 128,256, and 128 for the first until the last prediction layers, respectively. Furthermore, it uses a 2-stages strategy to increase the number of matched anchor

for more effective training.

As observed in the experiments, the first detection layer contains around 75% of anchors but over 99.8% belongs to background. It causes a large imbalance between positive and negative classes in training. Therefore, S3FD proposes to uses max-out background prediction to solve this problem. Specifically, each prediction layer contains 1 foreground and n-background predictions. The maximum value of the background score is then selected as the final prediction score alongside the foreground score.

The feature pyramid network (FPN) was designed to boost detection performance on various scales (Lin et al., 2017). Detection of small-sized objects in the single-shot is performed at low-level layers. However, features in the low-level layers typically contain less rich information which may not adequate to describe the objects. Therefore, this method proposes to combine the low-level features at the early layer with the high-level features from the deeper layers.

The dual-shot face detector (DSFD) utilizes an FPN-like architecture to improve its performance (Li et al., 2019). It proposes a feature enhancement module (FEM) to combines the lower-level features with the higher-level features.

The dual-shot face detector (DSFD) utilizes an FPN-like architecture to improve its performance. It proposes a feature enhancement module (FEM) to combines the lower-level features with the higher-level features. The Feature map from a higher-level layer is typically smaller than the lower-level feature map. Therefore, the FEM performs a convolution and then upsampling to the Up-feature-map before combining it with the current feature map, augmented with a single layer convolution, via a dot product operation.

The FEM passes the combined features to a 3-branches network. Each branch uses dilated convolution kernels with different amount of stages. Outputs from all branches are then concatenated and then passed to its corresponding prediction layer.

DSFD method uses 2 stages of predictions in its training step. The first stage predicts the outputs from the basic features similar to the S3FD method. On the other hand, the second stage performs prediction from the enhanced features. Experiment

results show that this strategy produces better results compared to the same architecture trained with single-shot detection.

The WIDER dataset (Yang et al., 2016) provides a benchmark for various face detection methods. In this benchmark, there are more than 20 methods are competing with each other including the traditional models (Yang et al., 2014; Mathias et al., 2014; Viola and Jones, 2001) and the deep learning models (Chi et al., 2019; Najibi et al., 2017; Zhu et al., 2018; Cai et al., 2016). Most of the recent deep learning models outperform the traditional models by a large margin.

2.2 Face Recognition Network

The face recognition research consists of identification and verification. Face identification concerns about determining who is the person in a given image. Meanwhile, the face verification (also known as authentication), focuses on determining whether the identity from 2 images are the same or not. The verification task is a 1-to-1 matching test, while the identification task is a 1-to-many comparison task. Figure 2.5 illustrates the 2 types of face recognition approaches.

Deep learning is a powerful method to perform classification. With the availability of a sufficient amount dataset (i.e. face images and their corresponding identities), it could determine the identity of a given face. However, it comes with some drawbacks which make the system becomes impractical. First of all, it needs a large amount of data for each identity. Second, this kind of system does not scale well since new training on the whole dataset should be performed every time a new identity is inserted into the system.

A practical way for the scalable person identification system is using the clustering principle. Features from face images that correspond to the same identity should fall into a distinctive cluster. Every time a new identity comes, all features correspond to it should fall in a different cluster.

FaceNet is a CNN model that adapts the clustering concept for face identification (Schroff, Kalenichenko, and Philbin, 2015). It utilizes a training strategy via the triple loss function. It treats the CNN model as a feature extractor. Using the loss function, it trains the model to produce embedding features that close each other when they

come from the same identity and stay apart conversely.

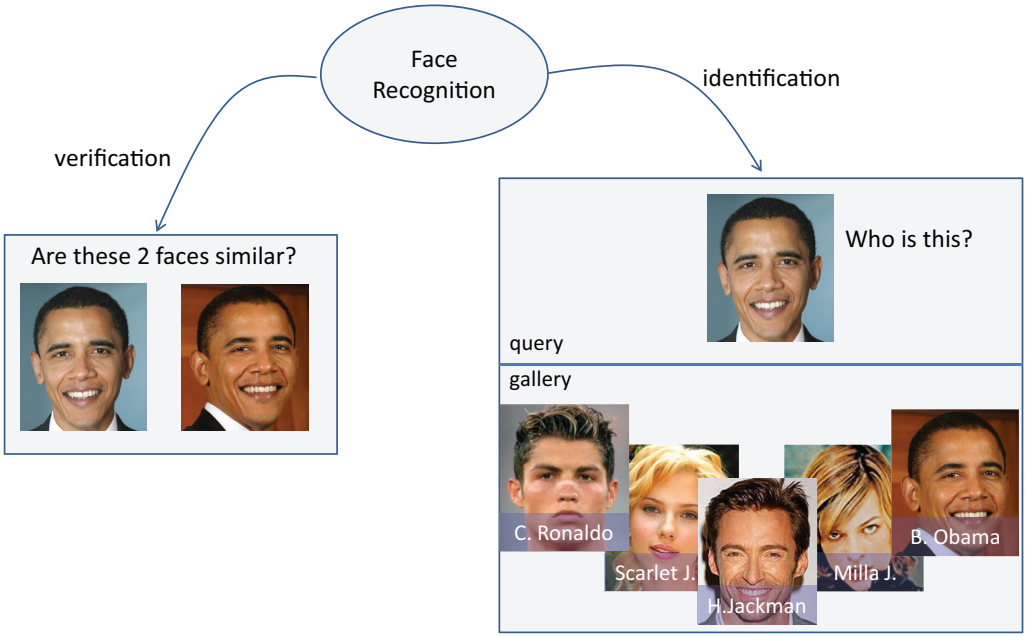

FIGURE 2.5: Sub-fields of the face recognition research.

In triplet loss, one set of training samples consists of 3 images, that form positive and negative pairs. It extends the siamese loss (Chopra, Hadsell, and LeCun, 2005) that enforces the distance from positive pair to be smaller compared to the negative pair. Furthermore, it employs a margin that strengthens the property of a smaller distance for positive pair compared to the negative pair.

Various works are proposed to increases the performance of face identification. Higher performance is achieved when the model is pre-trained to classify a set of faces before the triple-loss training (Parkhi, Vedaldi, and Zisserman, 2015). According to experiments, it delivers comparable results on training with less dataset.

Another remarkable extension is the center loss method (Wen et al., 2016). It enforces the embedded features to be as close as possible to the center of the cluster. If the embedding stays far away from its cluster center, a large penalty is enforced in the training step and vice versa.

Several other improvements are introduced (Deng et al., 2019a; Liu et al., 2017; Liu et al., 2018b; Wang et al., 2017; Wang et al., 2018). Among them, ArcFace (Deng et al., 2019a) shows outstanding performances in various benchmarks including the LFW (Kemelmacher-Shlizerman et al., 2016), CPLFW (Zheng and Deng, 2018), CALFW (Zheng, Deng, and Hu, 2017), YTF (Wolf, Hassner, and Maoz, 2011),

MegaFace (Kemelmacher-Shlizerman et al., 2016), iQiyi (Liu et al., 2018c)), and IJB dataset (Whitelam et al., 2017).

In ArcFace, embeddings are normalized so that all of them stay on the sphere surface. To enforce a large margin between identities, it uses an angular margin added to the angle between projection weights and the embedding. Therefore, the final embedding stays nearby the projection weights, leaving a large angular distance across different identities.

2.3 Inference Optimization for Deep Learning Model

A deep learning model requires a lot of computation to perform its prediction. As a result, it is not easy to deploy a model in low-performance devices or embedded computers that have limited computation power. Furthermore, a deep learning model often spent a lot of processing time even when deployed in a high-end GPU. This limitation restricts the model for a real-time system that requires a fast computation time.

Despite the availability of small-sized accelerators such as Intel NCS, NVIDIA Jetson Nano, and Google Coral (edge TPU) that have comparable performance to the high-end GPU in terms of inference time, a deep learning model is often too big to fit in these embedded devices or run to slow. In recent studies, speeding up the deep learning computation is gaining a lot of attention. Several approaches are including parameter pruning, quantization, and automatic architecture design which aims for a more efficient model.

2.3.1 Network pruning

Pruning is a way to remove one or more connections from the network architecture. It makes the network works faster during inference and stored more efficiently in the memory due to its smaller size.

Connection removal is not a trivial task. Some connections are important for retaining the prediction quality. Therefore, it is necessary to determine the importance of each connection before performing the pruning step.

As some connections are pruned, prediction quality tends to be decreasing. One way to retain the quality in the pruned network is by performing fine-tuning after a pruning step (Han et al., 2015).

As some connections are pruned, prediction quality tends to be decreasing. One way to retain the quality in the pruned network is by performing fine-tuning after a pruning step. According to the experiments, this strategy could effectively reduce the parameter of AlexNet (Krizhevsky, Sutskever, and Hinton, 2012) up to 50% without significant performance degradation. Furthermore, with iterative pruning and fine-tuning strategy, the experiment result shows that pruning with a 90% rate still retains the original performance of AlexNet.

The sequence of pruning and fine-tuning strategy shows that training a neural network with good initialization leads to a good performance. This property, especially in terms of network pruning, is formalized in the lottery ticket hypothesis (Frankle and Carbin, 2019). It states that a randomly-initialized dense neural network contains a sub-network (with good initialization) which can match the test accuracy of the original network after training for at most the same number of iteration.

A remarkable improvement in network pruning is proposed in the deep compression method (Han, Mao, and Dally, 2016). It improves the compression by employing a trained quantization which makes the parameters use a fewer bit compared to the original network. This method claims that the quantized training reduces the model size up to 27 times without loss of accuracy. However, practical usage is still an issue since it needs specialized hardware to deal with the non-standard quantized values.

2.3.2 Neural Architecture Search

Designing a robust deep learning model with few parameters is not a trivial task. It is a tedious job that requires a lot of trials and errors. A practical way to handle this task is by letting the computer to design a model with the requested constraints.

Various approaches in automatic architecture design are performed using evolutionary computation (Liang, Meyerson, and Miikkulainen, 2018) and neural networks (Zoph et al., 2018; Liu et al., 2018a)). Recently, the neural architecture search

(NAS) gains a lot of attention due to its robustness and fast delivery.

NAS is a way to utilize a neural network in designing a new model for a specific task with the given dataset, selection of modules, resource limitations, and other constraints. One of the successful attempts is NasNet Zoph et al., 2018. It utilizes a recurrent neural network (RNN) as a controller to select the connection of possible modules in its search space. Experiment results, with 2000 GPU hours of training, shows that the model generated by NasNet perform better than the man-made designs such as Inception (Szegedy et al., 2016; Szegedy et al., 2017) and ResNet (He et al., 2016).

The main drawback of NasNet is the requirement of huge computing power. Efficient NAS (ENAS) was proposed to solve this problem by utilizing parameter sharing between possible network architectures (Pham et al., 2018). This strategy allows for more efficient training and produces a comparable result at a faster time. According to the experiments, ENAS produces comparable results to NasNet in less than half a day of training, while the NasNet requires around 3-4 days of training time.

In recent studies, connections between the selection of modules across layers are determined using weights trained by back-propagation (Liu, Simonyan, and Yang, 2019; Dong and Yang, 2019). This scheme is possible because each module is differentiable and the output of each layer is modeled as a weighted sum of all outputs from all modules (Dong and Yang, 2019). The weights are trained so that one of the modules in a certain layer becomes close to 1 while other weights are close to zero. Furthermore, the overall architecture should produce good predictions which are reflected by its task-specific loss according to the dataset.

Chapter 3

High Performance Face Detection Network

The research in face detection has been evolved rapidly since the introduction of a boosted cascade method by Viola&Jones (Viola and Jones, 2001). Despite its simplicity and real-time applicability, this method suffers from a lack of robustness on the realistic scene under various conditions, as reported in the WIDER face detection benchmark (Yang et al., 2016).

Deep learning has been shaping the modern computer vision algorithms. It is adopted in most of the recent face detector methods as reported in the Wider benchmark Yang et al., 2016. This kind of method allows the system to extracts some useful features based on the available dataset. In contrast, the traditional face detectors require manual feature engineering (Mathias et al., 2014; Viola and Jones, 2001; Yang et al., 2014). Feature engineering is a tedious task that requires a lot of trial and error efforts. Since features are one of the important ingredients in a successful machine learning model, the deep learning-based methods achieve better performance by a large margin compared to the feature engineering-based methods when a large amount of dataset is available (Cai et al., 2016; Li et al., 2019; Najibi et al., 2017; Yang et al., 2016; Zhang et al., 2017; Zhu et al., 2018).

In the deep learning-based face detection methods, architecture design and training strategy play important roles in the performance improvement. In term of model design, the two stages object detection models Dai et al., 2016; Ren et al., 2015 and single shot detectors Liu et al., 2016; Redmon et al., 2016; Redmon and Farhadi, 2017 are influencing the architecture design. Meanwhile, other training strategies

are including data augmentation method Li et al., 2019; Zhang et al., 2017, multi-task learning Deng et al., 2019b; Wang, Yuan, and Yu, 2017, anchoring strategy Chi et al., 2019; Zhu et al., 2018, and context-aware training Li et al., 2019; Tang et al., 2018.

The work in this manuscript focus on two things. First, it aims to utilize the merit of multi-task learning strategy which is expected to helps the training step of a face detector network. Second, improving the processing speed through the parameter reduction approach.

3.1 High Performance Face Detection System with Attention-Guided Model

Multi-tasks learning is a way to improve the quality of a neural-network model. It allows a model to converts its input into a feature vector which is passed to several prediction modules at once. Therefore, multiple tasks are performed simultaneously using one backbone network.

In this work, the face detection model is coupled with an attention-map generator as illustrated in Figure 3.1. The system fetches an input image to a network that is based on the SSD model. In addition to the prediction of the face bounding boxes, this network is designed to predict a heatmap that corresponds to the location of faces in the input image.

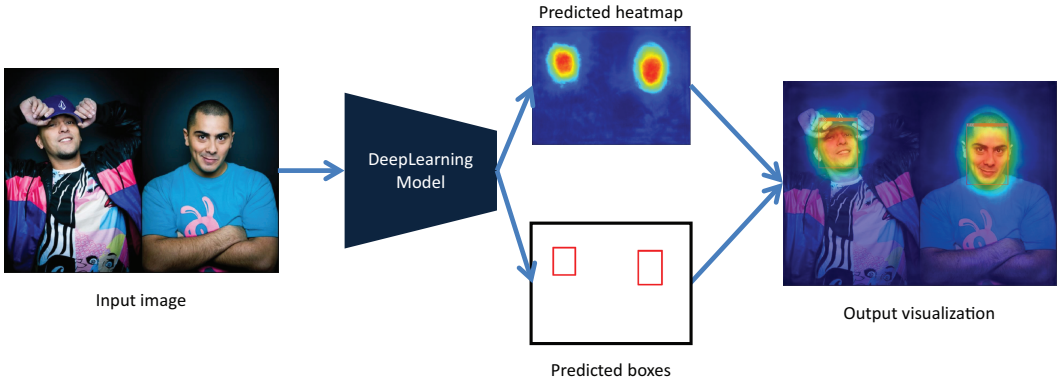

FIGURE 3.1: Overview of the proposed method.

3.1.1 System Architecture

The detector system in this work is designed based on the single-stage detector (SSD) model with a feature pyramid network module that allows the extraction of rich features for the higher resolution of the detection layer as depicted in Figure 3.2.

There are 6 layers of detector where each of them works for different scales of the face. The prediction starts at layer with stride 4 which corresponds to the quarter size of the input image. Since the feature in this layer is not rich enough, a feature enhancement module (FEM) enhances it as in (Li et al., 2019) by combining the high-level feature from the pyramid layers and low-level information from the current layer.

As a result, the prediction of face and non-face is derived by rich features that are expected to give a better representation of the face or background region. This kind of strategy is applied in all of the detection layers to encourage the utilization of both low-level and high-level feature representation.

The attention map is predicted by an extra convolution after the feature enhancement module which corresponds to the finest detection region (i.e. layer with stride 4). This map is incorporated with a purpose to help the training process and making the network focus on the region that has a high probability to contains a face. Optionally, it is useful to perform feature masking in each detection layer before the prediction of bounding box offset as shown in Figure 3.2.

In order to train the proposed model, a common detection loss which consists of classification loss and localization loss is utilized as formulated in (3.1). The classification loss (L_{cls}) is a cross-entropy loss between predicted logits and the classification ground-truth while the L_{loc} is Huber loss between predicted bounding box offset to its corresponding ground-truth as in SSD (Liu et al., 2016). Classification loss is calculated for all of the anchors while localization loss is only calculated for the positive anchors.

$$L_d = \frac{1}{N_{cls}} \sum_{m}^{N_{cls}} L_{cls}\left(c_i, \tilde{c}_i\right) + \frac{1}{N_{loc}} \sum_{m}^{N_{loc}} L_{loc}\left(o_i, \tilde{o}_i\right) \tag{3.1}$$

Meanwhile for the attention loss, the mean of the squared difference between ground-truth A and predicted mask $\tilde{A}$ is utilized as shown in (3.2). In this case, W

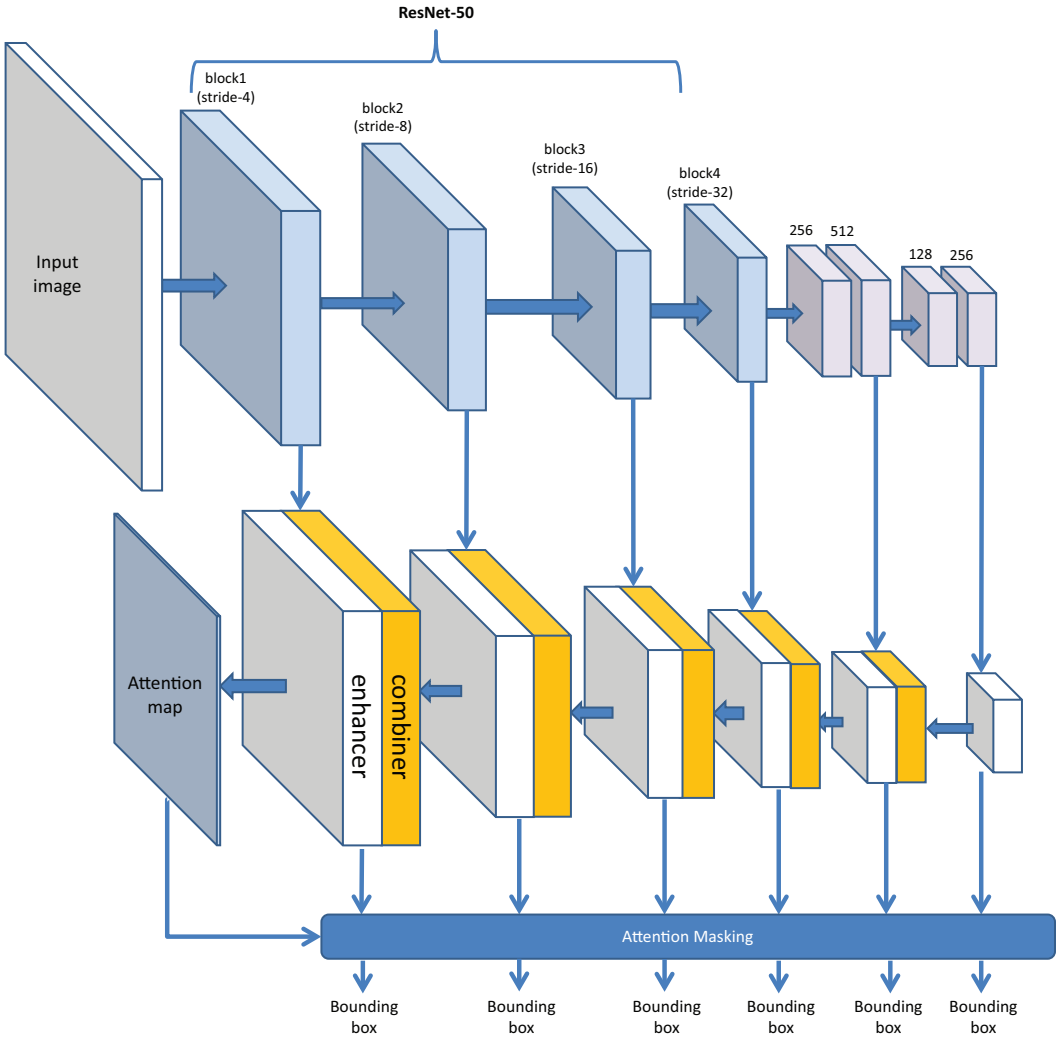

FIGURE 3.2: Architecture of the face detector network.

and H are the width and height of the attention map, respectively.

$$L_a = \frac{1}{WH} \sum_{i}^{H} \sum_{j}^{W} \left(A_{i,j} - \widetilde{A}_{i,j} \right)^2 \tag{3.2}$$

To fully train the model, multi-task loss utilized by combining the detection loss and attention prediction loss. As in Li et al., 2019, two kinds of detection loss is calculated for both first shot (from top stream in Figure 3.2) ad second shot (from 2nd stream in Figure 3.2). The final loss value is formalized as (3.3) where L_d^i is the detection loss for the i^{th}-shot of bounding box prediction.

$$L_{total} = L_d^{p1} + L_d^{p2} + L_a \tag{3.3}$$

3.1.2 Soft Attention Mask and Data Augmentation

The mask is designed as a multivariate Gaussian distributed values where its maximum is located in the center of a given face bounding box. It is formulated as (3.4) where σ_1 and σ_2 are set as half value of face bounding box width and height, respectively. Therefore, for each position at row r and column c in the ground-truth attention mask, the value is determined by an exponentially decayed distance between this location and the center of the bounding box (b_y, b_x) with a standard deviation σ_1 and σ_2.

$$A_{r,c} = e^{-\frac{1}{2\sigma_1\sigma_2}\left(r-b_y\right)^2(c-b_x)^2} \tag{3.4}$$

To train the proposed method, a dataset augmentation strategy similar to (Li et al., 2019) and (Liu et al., 2016) is utilized. Specifically, from a given image, one face is randomly selected and then a square region around it is sampled at random location to ensure the selected face is inside the cropped region while the size is set to make the face area falls within ratio nearby [16, 32, 64, 128, 256, 512]/640 compared to the cropped region. Furthermore, random padding is also added to make sure that the cropped region has a square size and generate samples with small faces in it.

FIGURE 3.3: Example of the data augmentation result.

Several data augmentation results are shown in Figure 3.3. On the left-most image, the sample was generated with padding. As a result, the augmented image contains small faces on it.

On the second image, some faces are not fully contained in the crop region because they are not selected as the main sample, the ground-truth corresponds to these kinds of faces are retained only if its original bounding box center is included in the crop region.

In all of the images, the attention mask is highlighting the face region and its

TABLE 3.1: Mean of Average Precision (mAP) of the Proposed Model at IoU 0.5.

Method	Evaluation Set		
	Easy	Medium	Hard
DSFD-ResNet50 (Li et al., 2019)	95.3	94.4	88.6
DSFD-ResNet50 (own impl.)	94.9	94.1	87.4
Proposed - Guided	95.7	**94.6**	87.9
Proposed - Guided w/ pyramid scaling	95.4	94.4	**88.8**
Proposed - Masked	**95.8**	**94.6**	87.9
Proposed - Masked w/ pyramid scaling	95.4	94.4	**88.8**

nearby. It is designed in this way to ensure that region nearby faces are contributing to the bounding box detection result.

3.1.3 Experiments and Results

To evaluate the proposed method, the WIDER dataset Yang et al., 2016 is used to train the model. Augmentation and generation of attention mask ground-truth are performed as specified in section 3.1.2. The model was trained with batch size set to 32 during 150k iteration using the gradient descent optimizer (momentum=0.9) and L_2 weight regularization scaled to 5e-4. The learning rate is set as 1e-3 and dropped by a factor of 0.1 at 80k, 100k and 120k of training steps.

In this study, the ResNet50 is utilized as a backbone network to make it comparable with the experiments in Li et al., 2019. To preserve the fairness of performance comparison, the DSFD-ResNet50 model is re-implemented as the baseline for this experiment. As shown in Table 3.1, the performance between our implementation is similar to the original result in Li et al., 2019. It should be noted that we did not test our implementation of DSFD-ResNet50 using the pyramid scaling test as utilized in its original paper. Therefore, there is a 1% performance gap in the hard evaluation set compared to the result reported in the original paper.

In the original pyramid scaling scheme (Li et al., 2019), an input image is tested at 0.25, 1, 1.25, 1.75, and 2.25 of the original size. Meanwhile, in our implementation, the scale is started from 0.5, then doubled in several steps until the GPU is approximately full. Additionally, evaluation of the flipped image is also performed on both types of tests.

As shown in Table 3.1 the pyramid scaling strategy boosts the mAP score on the

hard evaluation set, but unfortunately it is lowering down the score in both easy and medium sets. However, the overall results are still better compared to the baseline. This is showing the proof that our proposed method is useful to boost the performance of the baseline which is not utilizing attention mechanism in either training or inference phase.

Performance evaluations against the state of the art methods are also presented in Figure 3.4. It summarizes the comparison of various methods in the easy, medium, and hard validation set from top to bottom respectively. As reflected in all of the evaluation set, the proposed method delivers competitive results compared to the other state of the art methods. It should be noted that the deep learning-based models are outperforming the feature-engineering based method such as ACF by a large margin as shown clearly in Figure 3.4.

As reflected in all of the evaluation set, the proposed method delivers competitive results compared to the other state of the art methods. It should be noted that the deep learning-based models are outperforming the feature-engineering based method such as ACF by a large margin as shown clearly in Figure 3.4.

Several examples of qualitative results are presented in Figure 3.5. The predicted attention map produces acceptable results as designed to emphasize the regions nearby faces. Often, it highlights incomplete face regions such as the one on the top-right side of the first image and the same goes in the second image.

3.2 Efficient Face Detector

Deep learning achieves a state of the art performance in various tasks. It typically requires a lot of data fed into complex architecture repeatedly during the training process. Despite its potentially satisfying results, a deep learning model is often over-parameterized.

A large-sized network often contains redundant connections. It leads to unnecessary computation and slow computation time. These properties are undesirable for production-ready industrial products.

The current high performer face detectors are built on top of computationally demanding models, such as VGG16, Resnet101, or Resnet152. A typical SSD model as

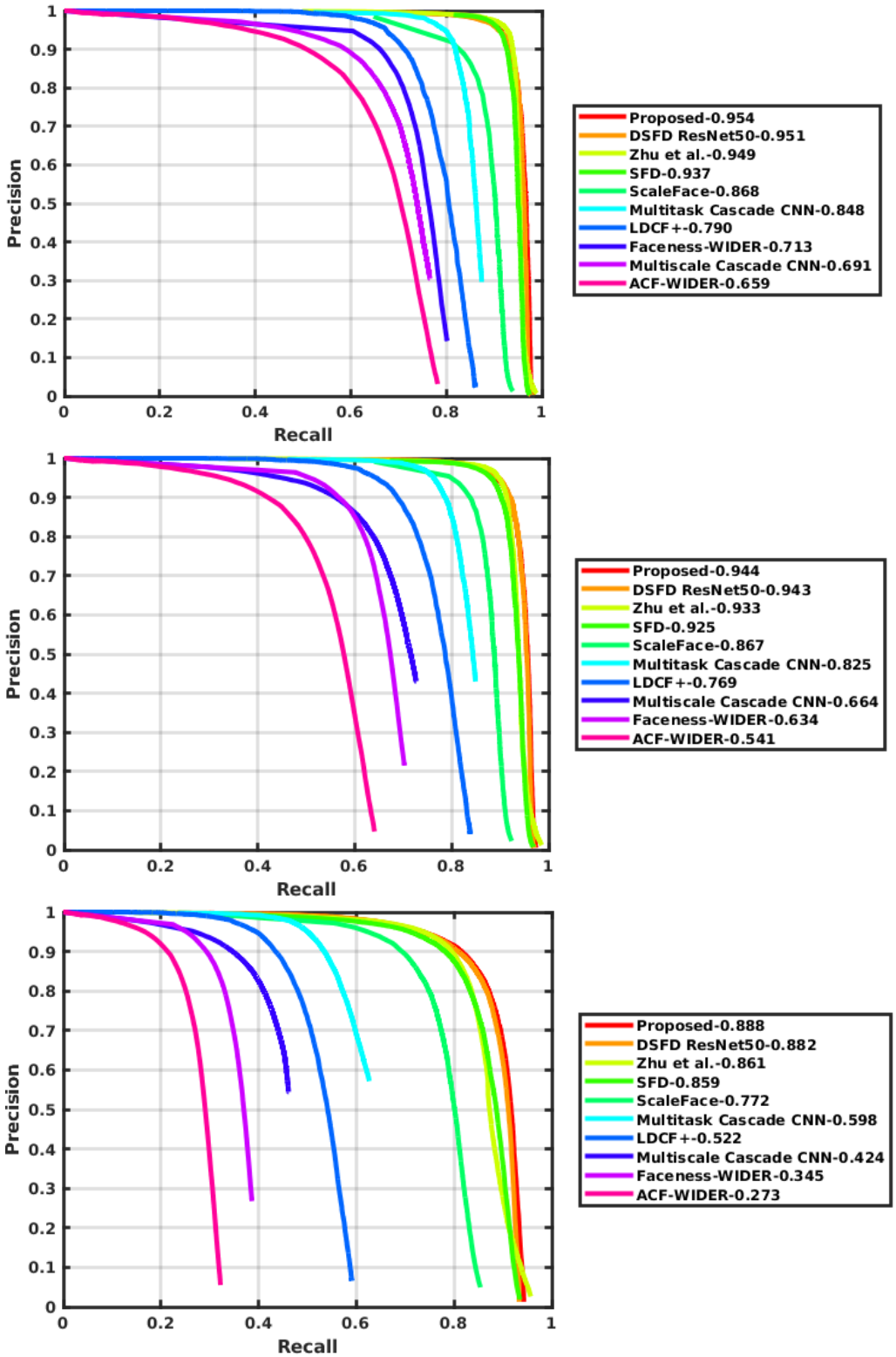

FIGURE 3.4: Overview of the proposed method. A deep learning model is designed to predict heatmap of faces as well as their corresponding bounding boxes.

an example, with the VGA input image it requires approximately 30ms of processing time on GPU.

FIGURE 3.5: Examples of the successful detection result using the proposed method. Best viewed in color or monitor.

A model with 30ms of inference time is suitable for a real-time application. However, face detection is only a first step of processing in various applications. Therefore, a real-world system requires a faster face detector to accommodate other tasks to run within the real-time processing constraints.

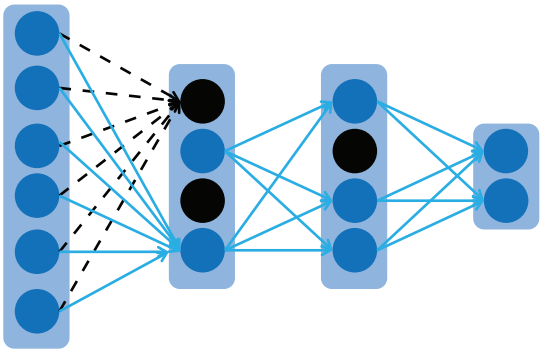

FIGURE 3.6: Illustration of a pruned network. Black dashed-lines corresponds to a dropped node in the next layer.

Model pruning is an approach to reduce the processing time of deep learning architecture. It removes unnecessary connections in the over-parameterized networks as illustrated in Figure 3.6.

Some connections in a network may contain very small weights. When all of the weights for a node in the deeper layer are close to zero, the activation value is getting smaller. A node with a very low activation value is insignificant to the activation value in the next layer. Therefore it is also insignificant to the final result. Hence, removing this kind of node from a deep learning model will not make the overall performance dropped significantly.

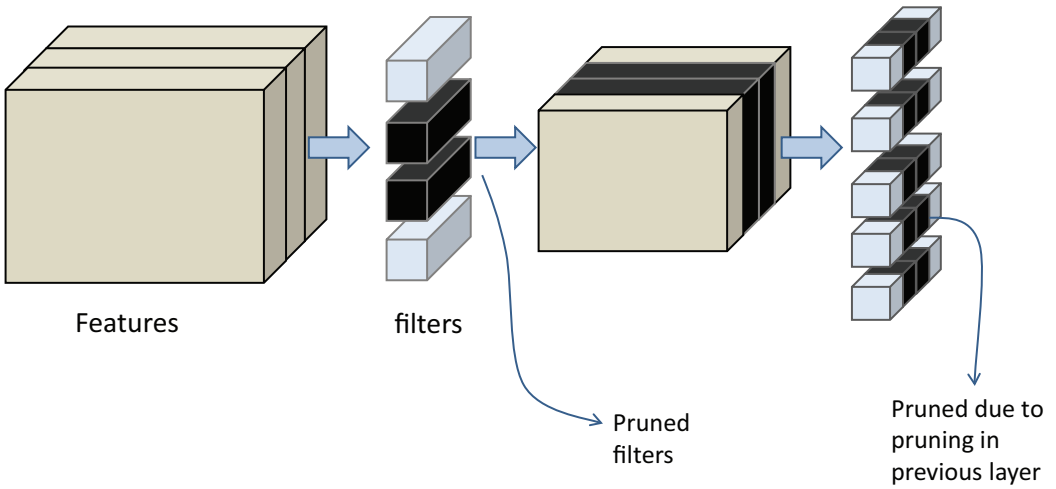

FIGURE 3.7: Channel-wise pruning strategy. Removed weights and feature channels are black-colored.

The work in this research is focused on the filter pruning. It is especially useful for the convolutional module which is the main component in a face detector network. It removes the filters with low importance score and retains the useful filters. As a result, the model size is getting smaller in a non-sparse function. Therefore, it is possible to deploy the pruned model in any conventional hardware.

Figure 3.7 shows the illustration of the filter pruning. Convolution operations on feature F at layer i (F_i) by n-filters (W_i) produce a new feature map (F_j) with n-channels.

In the pruning step, importance scores are calculated for each filter $W_{i,k}$. This score is calculated based on (3.5) which represents the sum of absolute values from each element of $W_{i,k}$.

$$s_{i,k} = \sum |W_{i,k}| \tag{3.5}$$

Top-k filters are retained based on their importance score while the rest are removed (black color). As a consequence, k-channels are remaining in its output feature map (F_j).

As shown in Figure 3.7, pruning operation in F_i affect the size F_j. Since there are k-channels are remaining after pruning of F_i, each filter in F_j should also consist of the same amount of channels. Therefore, pruning is performed in F_j in a channel-wise fashion.

3.2.1 Pruning Strategy

A network pruning is performed by removing the least important filters based on their importance score. However, directly removing a large amount of filter often leads to bad network performance.

In this work, an iterative pruning strategy is utilized to reduce the parameters of a given face detector. Algorithm 1 and Figure 3.8 shows the overall steps of the pruning.

Firstly, a standard training procedure is performed on the face detector model

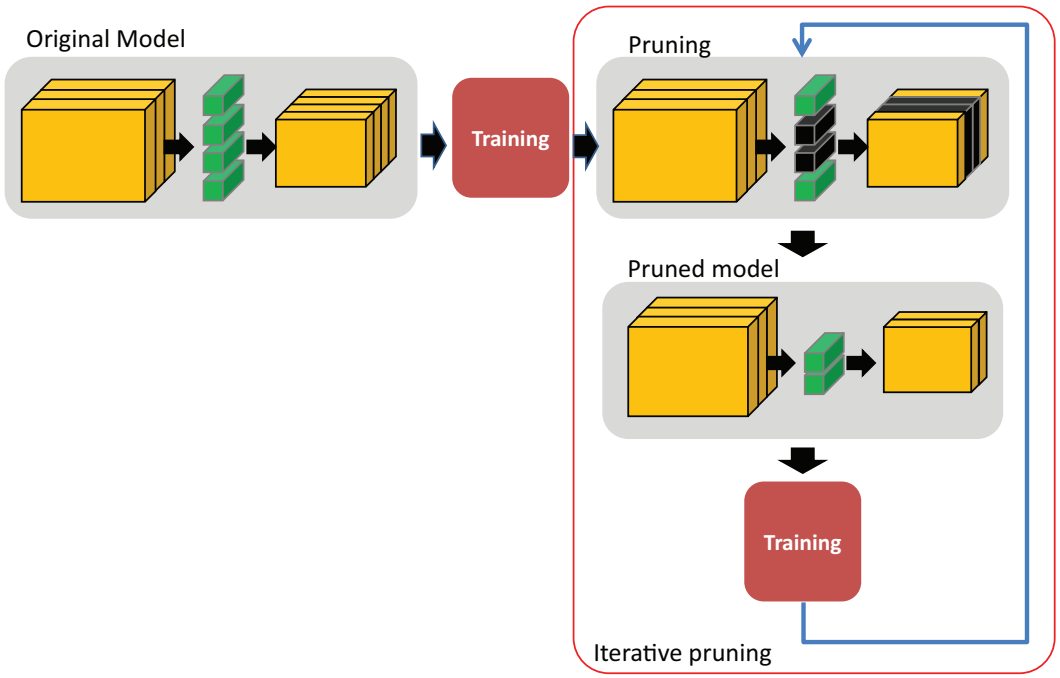

FIGURE 3.8: Illustration of the iterative pruning strategy.

using gradient descent and backpropagation procedures. Weight updates are performed numerous times to ensures that the model is well trained. After the training is converged, the pruning strategy is ready to be started.

Iterative pruning is performed for several steps by a few amounts of pruning ratio (τ). Firstly, it calculates the importance scores of each weight filter. It then selects the top-k filters based on the pruning ratio and importance score and removes all the remaining filters. Removing a few amounts of filter leads to a performance drop. Hence, a fine-tuning process is performed to recover the model performance.

3.2.2 Experiments

To test the usefulness of the pruning strategy, experiments are performed on the single-shot scale-invariant face detector (S3FD) model on the wider dataset for training and evaluation. The S3FD model is trained from scratch with configuration as suggested in the original paper. This base model achieves similar performance to the result in the original paper as shown in Table 3.2.

Iterative pruning is performed 9 times to obtains several models at various pruning rates. In each step, 10% of the original model parameters are removed. Fine-tuning is then performed for 60k steps with learning rate 1e-4 for the first 20k and

Algorithm 1: Iterative Pruning

Input : model weights W
Input : pruning ratio τ
Input : maximum pruning iteration n
Output: pruned weights $\hat{W}$

/* Training the model */
while *not converged* **do**
  $W \leftarrow W + \eta \frac{\partial L}{\partial W}$
end
/* Iterative pruning */
$t \leftarrow 0$ // pruning step
$\hat{W} \leftarrow W$
while $t < n$ **do**
  $t \leftarrow t + 1$
  $s_{i,k} \leftarrow \sum \left|\hat{W}_{i,k}\right|$ // calculate the importance score
  $\hat{W} \leftarrow P\left(\hat{W}, \tau, s\right)$ // filter pruning
  while *not converged* **do**
    $\hat{W} \leftarrow \hat{W} + \eta \frac{\partial L}{\partial \hat{W}}$ // Fine tuning
  end
end

then dropped to 5e-5 until step 40k and finally 1e-5 until the end of the training step.

As shown in Table 3.2, there is no significant performance drop in all of the model with different pruning ratio. For the model with a 50% pruning rate, the performance drops are 0.5, 0.5, and 0.4 for the easy, medium, and hard validation data, respectively. In this model, the model complexity is dropped to 24.73% relative to the non-pruned model while the processing speed is increased by 1.6 times.

Figure 3.9 illustrates the performance drop for each pruning rate. The performance is quite stable until approximately 50% of the pruning rate and after that rapidly declined. We argue that this significant drop is caused by the removal of useful filters in the few last layers of the model. However, a more comprehensive evaluation should be carried out and we left this task as the future work.

TABLE 3.2: Peformance comparsion of the pruned S3FD models.

Pruning Rate	mAP			Complexity	Runtime	Speed
(%)	Easy	Medium	Hard	(GFLOP)	(ms)	(fps)
0	93.7	92.5	85.1	258.26	31	32.26
10	93.6	92.5	85.4	208.21	28.99	34.49
20	93.5	92.2	84.5	164.8	26.13	38.27
30	93.7	92.5	85.3	126.03	23	43.48
40	93.6	92.3	84.6	92.82	21.56	46.38
50	93.2	92	84.7	63.89	19.07	52.44
60	93.3	91.8	83.9	41.49	16.13	62.00
70	92.7	91.1	82.5	23.52	15.44	64.77
80	89.7	87.3	75.6	10.48	13.57	73.71
90	84.3	80.9	64.4	2.69	12.11	82.58

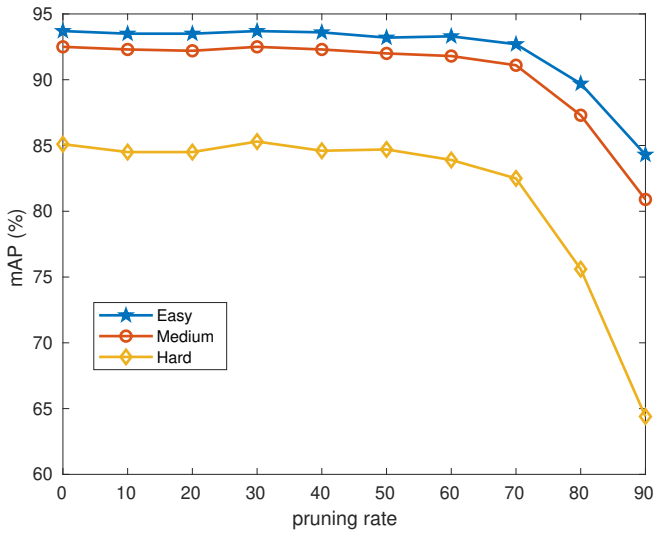

FIGURE 3.9: Effect of pruning rate to the system performance.

Chapter 4

Efficient Face Recognition Network

The Embedding-based models are the most popular approach to recognize face identity in a given image. Most of these models use deep learning as a feature extractor. Therefore, its popularity is getting higher by the availability of various deep learning accelerators such as GPU and other light-weight devices like Intel NCS, NVIDIA Jetson, and Google's Coral.

A deep learning model is a feature transformer. Figure 4.1 shows the 2D features from the MNIST dataset extracted using a model trained with a softmax classifier. This classifier, together with the cross-entropy loss, enforces the network to produce features aligned to its corresponding class weight vector. As a result, features are forming a star-like structure, as shown in Figure 4.1 top-left. Therefore, image features from the same class lie in one cluster.

The face verification step uses this clustering principle to determines whether faces are from the same identity or not. If they belong to the same person, then their corresponding embedding feature lies close to each other.

The features extracted from a deep-learning model form distinct clusters for each trained class. It is due to the training process tries to align the embedding features with their corresponding class-weight as illustrated in Figure 4.2. The prediction of a deep learning model (in this case is specific for the classification model) is determined by the maximum score corresponds to each predicted class label. This score is calculated based on the projection value of the embedding feature to the class-weights ($w_i^T x$). Therefore, the prediction is correct whenever condition (4.1) is met. In this case, the logit score ($w_i^T x$) that corresponds to the correct prediction of class i should be larger than the logit score to the other classes. Therefore, the only way of

condition (4.1) to be satisfied is when the embedding features lie aligned with their corresponding class-weight.

$$w_i^T x < w_j^T x; \forall j \neq i; j \in \{1, 2, ..., N\} \tag{4.1}$$

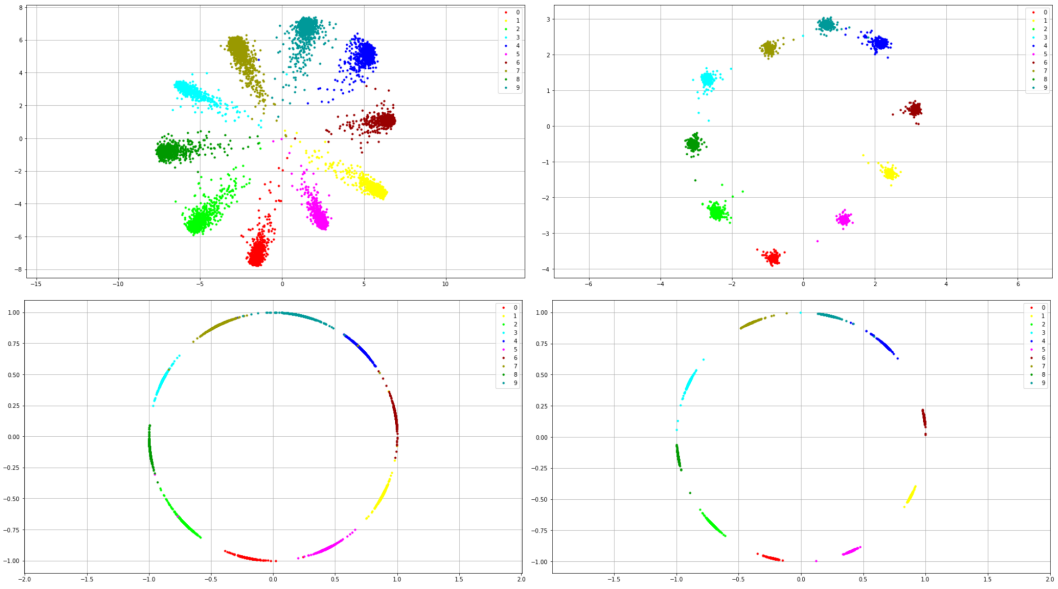

FIGURE 4.1: Embedding features extracted using a network trained on MNIST dataset using different types of loss functions. Left: softmax-loss, right: center loss, top: un-normalized, bottom: normalized. Best viewed in display.

Using a geometric distance to compare the face embedding similarity often leads to problems, especially for the basic softmax model. Features from different identities lie on the different clusters, but their distance may close each other, while features from the same identity lie farther. The inter-class failures are mostly happening with features lie nearby origin. On the other hand, intra-class failure happens with features a cluster where one of them lies nearby origin, and the other one lies nearby the tip of a cluster.

One way to improve the clustering quality is by using extra supervision such as center loss. It enforces the embeddings to come closer to their cluster center by minimizing the distance between each point to the cluster center. In every training step, cluster centers are updated dynamically. Therefore, embeddings are forming a more compact cluster compared to the standard softmax-loss result. Figure 4.1 top-right shows an example of the clustering result. Each cluster forms a circle-like blob instead of a long oval-like shape.

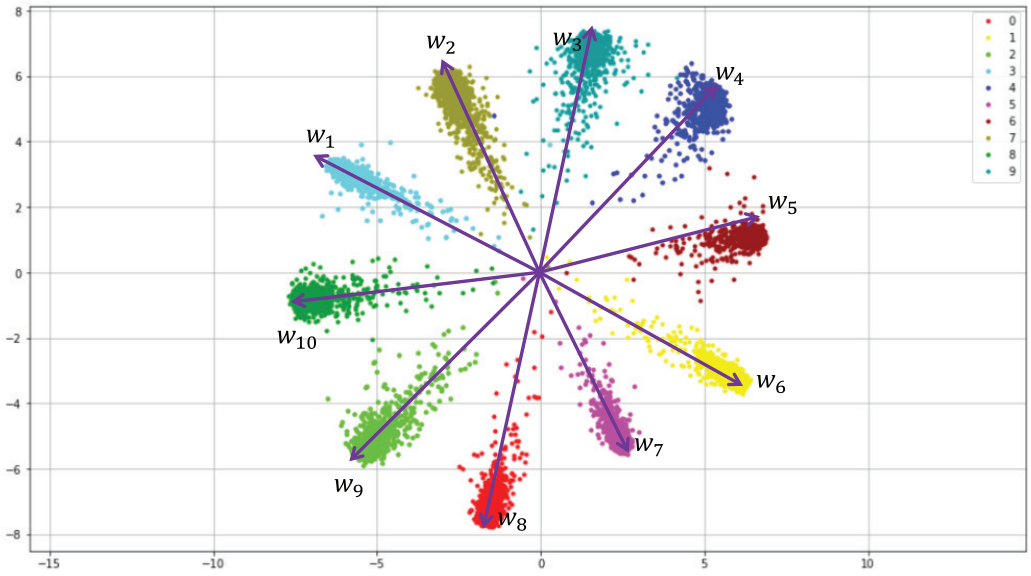

FIGURE 4.2: Illustration of the relation between embedding features and their corresponding class-weight. Best viewed in display.

Recent methods utilize l_2 normalization to the embedding features. This leads to easier feature comparison using angle-based distance which corresponds to the dot product between features. Figure 4.1 bottom shows the normalized embedding features for both softmax cross-entropy loss and center loss, respectively.

In general, embedding features trained using softmax cross-entropy loss forms less separable clusters compared to the embedding extracted from a model trained using center-loss. By applying angle based margin, the separability between clusters in normalized features is improved.

If there is a test face in which its identity is included in the training data, then its embedding vector lies on the correct cluster. If there is a test face with an identity that not available in the training data, then it should go to a new location, outside any cluster. This property is based on the assumption that a face is a combination of several other faces. As an example, the embedding feature of a child lies between the embedding features from its parent's faces.

4.1 Fast Inference using Pruned Model

The key to a high-performance face recognition network lies on 3 main components, backbone network, dataset, and training strategy. Among these components, the

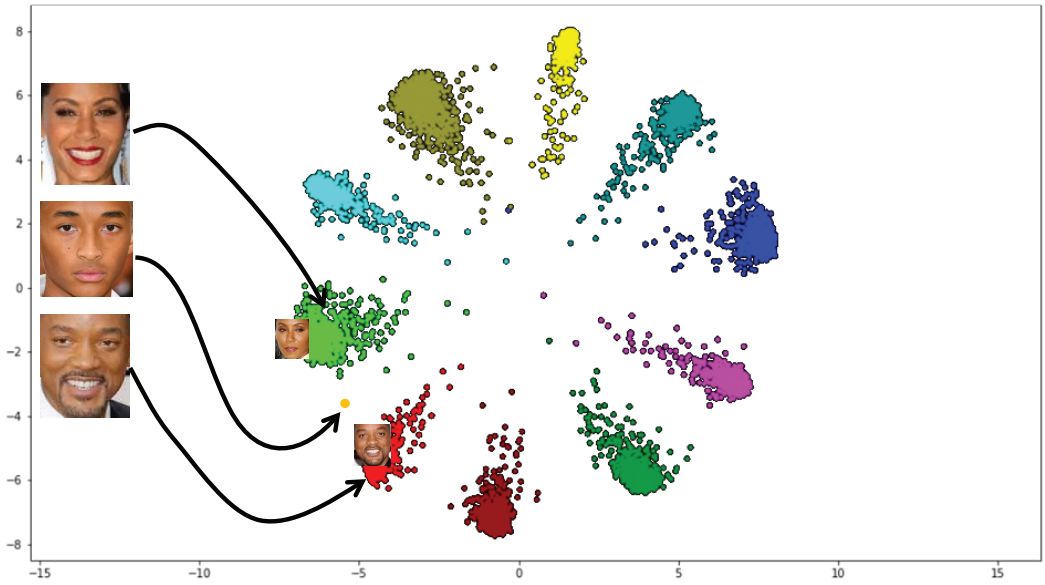

FIGURE 4.3: Illustration of the face identification using embedding features. Best viewed in display.

backbone network becomes a bottleneck in real-world applications due to the hardware limitation. Most of the state of the arts employ a backbone with heavy computation. Therefore, this work focuses on reducing the computation of a face identification model without compromising its performance.

Several ways to reduce the computation cost of a deep learning model are including parameter quantization, distillation, and model pruning. Quantization limits the numerical precision of a model using a lower bit representation. Theoretically, it could perform 4 times faster if 8 bits of precision are used instead of the standard 32 bits format. However, it needs specialized hardware or tweaks to make to model run on the real environment.

On the other hand, distillation is a common technique to merge an ensemble of larger models into a single model. The experiment on using it to make very small models that work in real-time is still limited.

Model pruning is another approach to reduce the number of parameters from a given model. In a conventional pruning, a node is dropped whenever its corresponding weights become zero. It leads to a sparse model which also requires specialized hardware. Therefore in this work, we are focuses on channel-wise pruning, as discussed in section 3.2.1.

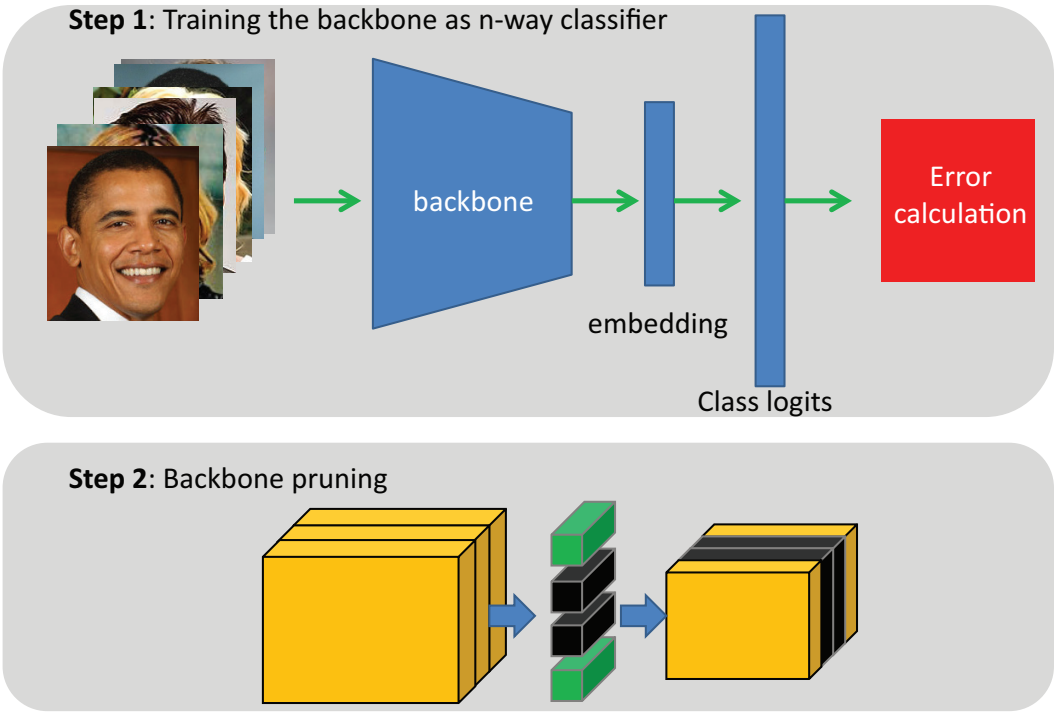

FIGURE 4.4: Pipeline for building the face recognition network.

Figure 4.4 explains the proposed steps to build an efficient face recognition network. Firstly, a backbone with extra layers to predict embedding features is trained using a dataset with a large number of identities. This training aims to perform n-way classification where n is the number of identities in the dataset.

After the network is converged, the pruning step is performed to simplify the model. An iterative pruning strategy is utilized as discussed in section 3.2.1. In summary, scores of the model filters are selected according to their rank. Then a portion of them is deleted, producing a simpler model. This pruned model is then re-trained (fine-tuning) until convergence. To ensure the model reaches the desired pruning rate, the pruning and fine-tuning process is repeated for several times.

4.2 Experiments and Results

To evaluate the effect of pruning on the face identification network, we prune a model with VGG16 as its backbone. Two convolution modules are added with a drop out layer in between to extract the embedding vectors. This model is trained using the CASIA dataset Yi et al., 2014. This dataset contains 10572 identities on approximately half a million images.

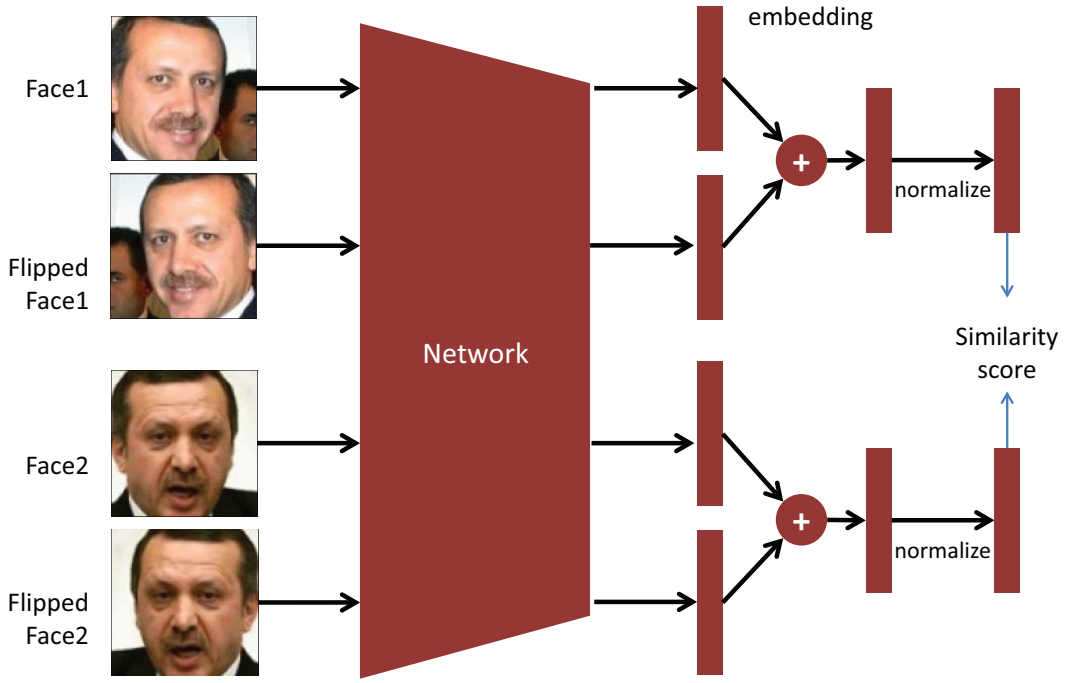

FIGURE 4.5: Similarity score calculation.

The model is initially trained for 100k steps with learning rate is set from 1e-3 and dropped by a factor of 10 at 30k steps. An SGD optimizer is utilized to train the model with momentum 0.9 and batch size 512.

Iterative pruning is then performed on the trained model with a 10% ratio in each round. This ratio is not set based on the original model but from the model at the previous pruning round. Therefore, the absolute pruning ratios are 10, 19, 27.1 34.39 and so on as shown in Table 4.1.

For evaluation, 6 dataset benchmark are utilized, including Labelled Face in the Wild (Huang et al., 2007), Celebrities in Frontal-Profile(Sengupta et al., 2016), Cross-Age LFW (Zheng, Deng, and Hu, 2017), Cross-pose LFW (Zheng and Deng, 2018), and AGEDB-30 (Moschoglou et al., 2017). Images are cropped using the MTCNN face detector and resized into 112x112 pixels.

The Labelled Face in the Wild (LFW) dataset consists of 5749 celebrities in 13233 images. The images are available as 250x205 pixel JPEG files where most of them are color images. The celebrities in Frontal-Profile (CFP) is available in two sets. The first set contains frontal to profile pairs (CFP-FP) while the other contains frontal to frontal pairs (CFP-FF). Most of the images are clear and in high quality. The Cross-Age LFW (CALFW) and Cross-Pose LFW (CPLFW) are collected from the

LFW dataset. CALFW contains image pairs from different ages while the CPLFW are from different poses. The AGEDB-30 dataset is quite challenging dataset since it contains images from a wide variety of age range. The image qualities are varying and most of them are in grayscale.

Figure 4.6 and 4.7 shows some examples of positive and negative pairs from the LFW and CA-LFW datasets. According to these two figures, the samples from CA-LFW are more challenging. For example, the face appearances of positive pairs in CA-LFW are quite different due to the age variation. In contrast, faces in the positive pairs of LFW datasets are quite similar since the age difference is not considered.

FIGURE 4.6: Example of positive pairs from the LFW and CA-LFW dataset. Re-drawn from the original paper (Zheng, Deng, and Hu, 2017).

Pruning is performed iteratively where each step removes 10% of the network filters. Figure 4.8 shows the evolution of channel size in each step of the VGG16 architecture. It should be noted that the number of convolution layers in Conv4 and Conv5 are the same. Therefore, both of them are represented as a single curve in the figure. As shown in the right part of Figure 4.8, the parameters in the last layers (Conv3, Conv4, and Conv5) are rapidly decreasing at pruning rate larger than 70% while the other earlier layers are decreasing in a slower pace

To evaluate the model, embeddings are extracted for each pair of the validation

FIGURE 4.7: Examples of negative pairs from the LFW and CA-LFW dataset. Re-drawn from the original paper (Zheng, Deng, and Hu, 2017).

dataset. Additionally, embeddings from their corresponding flipped faces are also collected. The embedding vector from the same an input image and its flipped version are then combined as illustrated in Figure 4.5. Therefore, there are 2 embedding vectors for each face.

Table 3.2 shows the experiment results for pruning at each round of pruning. Several strategies are tested to check which kind of combinations gives the best results. In total there are 4 types of combinations are considered including addition, addition and then normalization, concatenation, and concatenation with normalization.

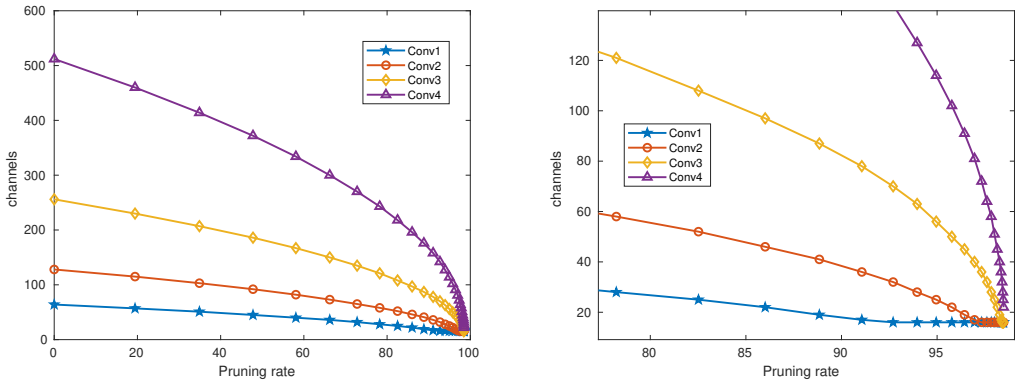

FIGURE 4.8: Evolution of the channel size across different pruning rate. Best viewed in display.

According to the experiments, the addition strategy gives poor performance compared to the 3 other methods. It gives 97.08% of accuracy in the non-pruned model while the others are at 98%. Its performance is getting worse as the pruned size getting smaller. It performs at 93.60% of accuracy at 81% of pruning rate while the others are still at 96% band or more.

TABLE 4.1: Performance evaluation of at different level of pruning rate on LFW dataset.

step	add	add norm	concat	concat norm	pruning rate
0	97.08 (0.00)	98.57 (0.00)	98.45 (0.00)	98.47 (0.00)	0.00
1	97.45 (0.37)	98.95 (0.38)	98.88 (0.43)	98.98 (0.52)	10.00
2	97.30 (0.22)	98.75 (0.18)	98.63 (0.18)	98.68 (0.22)	19.00
3	97.02 (-0.07)	98.87 (0.30)	98.75 (0.30)	98.82 (0.35)	27.10
4	96.72 (-0.37)	98.82 (0.25)	98.62 (0.17)	98.65 (0.18)	34.39
5	96.23 (-0.85)	98.52 (-0.05)	98.50 (0.05)	98.37 (-0.10)	40.95
6	96.02 (-1.07)	98.25 (-0.32)	98.28 (-0.17)	98.33 (-0.13)	46.86
7	95.92 (-1.17)	98.35 (-0.22)	98.37 (-0.08)	98.45 (-0.02)	52.17
8	95.48 (-1.60)	98.30 (-0.27)	98.38 (-0.07)	98.40 (-0.07)	56.95
9	95.45 (-1.63)	98.07 (-0.50)	98.07 (-0.38)	98.10 (-0.37)	61.26
10	94.98 (-2.10)	98.23 (-0.33)	98.13 (-0.32)	98.27 (-0.20)	65.13
11	94.98 (-2.10)	98.07 (-0.50)	98.07 (-0.38)	97.95 (-0.52)	68.62
12	94.42 (-2.67)	97.72 (-0.85)	97.83 (-0.62)	97.72 (-0.75)	71.76
13	94.47 (-2.62)	97.73 (-0.83)	97.65 (-0.80)	97.67 (-0.80)	74.58
14	94.07 (-3.02)	97.55 (-1.02)	97.47 (-0.98)	97.42 (-1.05)	77.12
15	94.15 (-2.93)	97.18 (-1.38)	97.35 (-1.10)	97.38 (-1.08)	79.41
16	93.60 (-3.48)	97.05 (-1.52)	96.60 (-1.85)	96.65 (-1.82)	81.47

Pruning with concatenation, concatenation with normalization, and addition with normalization shows satisfying results. Overall, the performance drops to 98% of its non-pruned model at almost every pruning rounds. This property still holds until the model size is around 20% of its original size. However, pruning with addition followed by normalization is more preferable since it performs faster at the score calculation time due to its lower embedding size (512 vs 1024). Therefore, the subsequent experiments are conducted with addition followed by concatenation strategy.

Figure 4.9 summarize the effect of iterative pruning on the verification performance on the 6 evaluation benchmarks. In the LFW and CFP-FF dataset, the pruned model retains its performance up to 70% pruning rate and then decreasing slowly until around 90% of pruning rate. It achieves high scores on both benchmarks since these 2 dataset contains comparison for frontal-to-frontal pairs.

The evaluation scores on frontal-to-profile pairs are relatively lower compared

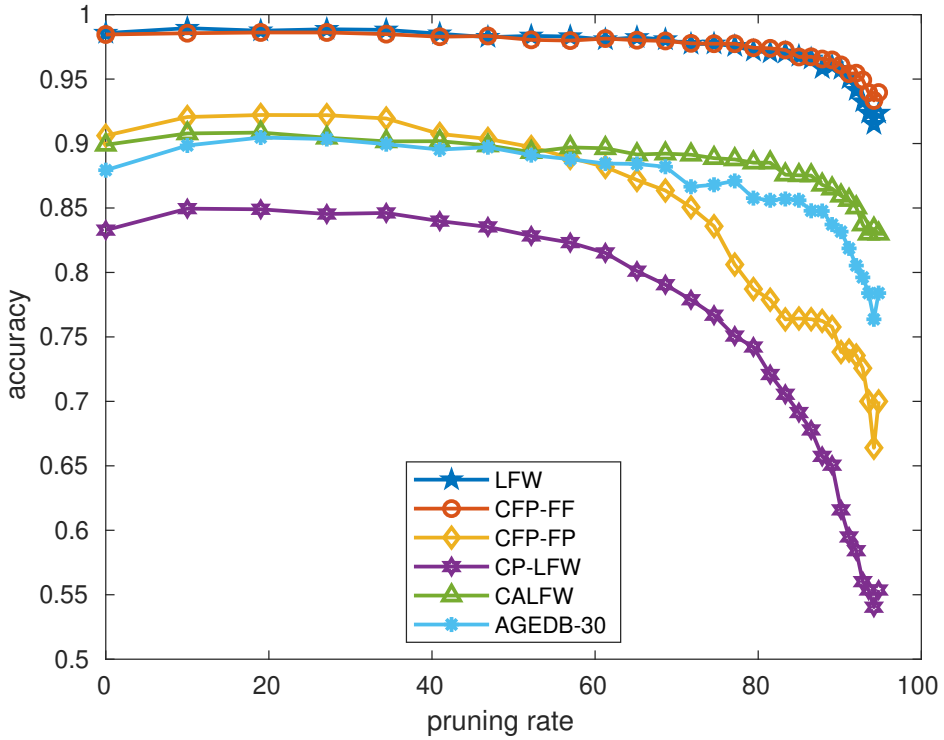

FIGURE 4.9: Performance of the Pruned VGG16 model on various benchmarks. Best viewed in display.

to the frontal-to-frontal pairs. This problem is more difficult due to the different appearance in the image pairs. As shown in Figure 4.9, the evaluation scores are peaked at around 92% on CFP-FP dataset and 85% on CP-LFW dataset. Similar results are obtained for cross-age verification where the scores are peaked at around 90% for both CALFW and AGEDB-30 benchmarks.

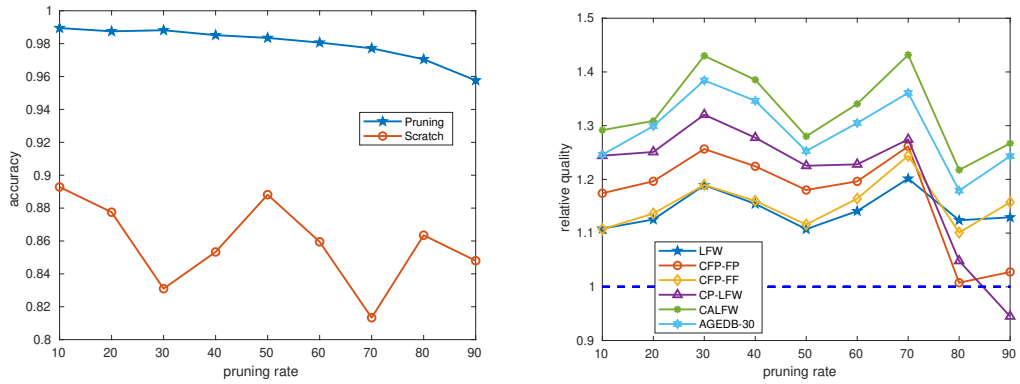

FIGURE 4.10: Performance comparison between iterative pruning and random initialization. Best viewed in display.

To proof that iterative pruning is an effective way to reduce the model parameters, experiments to compare this method with decreasing model parameters and

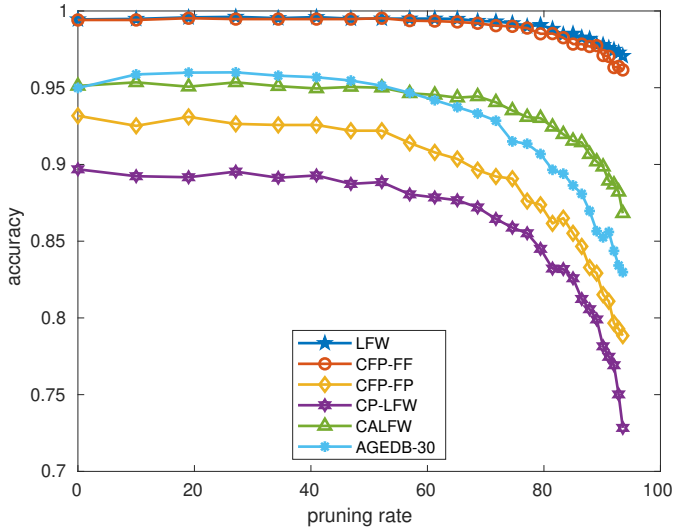

FIGURE 4.11: Performance of the pruned ResNet50 model on various benchmarks. Best viewed in display.

direct training on a model with the same number of parameters is performed. Figure 4.10 shows the performance comparison between iterative pruning and training from scratch. Iterative pruning derives the idea of fine-tuning which is essentially retaining the information from the previous training course. Therefore, when a few parameters are removed from the model, its performance is not decreasing rapidly. This performance drop is then recovered by the fine-tuning steps. On the other hand, training with random initialization on equivalent models produces lower scores compared to the iterative pruning strategy. This complies with the lottery ticket hypothesis (Frankle and Carbin, 2019) and shows that a network with better initialization leads to better performance.

Further experiments are conducted with the ResNet50 backbone. The network is trained using the MS1M dataset for 180K steps in one training session. The learning rate starts at 1e-2 and divided by 10 at step 100K and 160K. Other settings are the same as the experiment utilizing VGG16 backbone.

Figure 4.11 shows the effect of pruning rate on various dataset benchmarks. The performance is dropping slowly starting from 70% of pruning rate for LFW and CFP-FF dataset and drop quickly starting from 60% of pruning rate on 4 other benchmarks. However, this model achieves better performance compared to the model with VGG16 as its backbone in general.

TABLE 4.2: Performance comparison against state-of-the-arts models

Method	LFW	CALFW	CPLFW
Human-Individual (Zheng, Deng, and Hu, 2017)	92.27	82.32	81.21
Human-Fusion (Zheng, Deng, and Hu, 2017)	99.85	86.5	85.24
LBP-Euclidean (Zheng, Deng, and Hu, 2017)	74.68	62.37	-
LBP-Mahalanobis (Zheng, Deng, and Hu, 2017)	75.05	66.83	-
LBP-SVM (Cortes and Vapnik, 1995)	76.68	65.27	-
LBP-ITML (Davis et al., 2007)	82.37	68.82	-
LBP-KISSME (Koestinger et al., 2012)	77.57	67.87	-
CenterFace (Wen et al., 2016)	98.75	85.48	77.48
SphereFace (Liu et al., 2017)	99.27	90.30	81.4
VGGFace2 (Cao et al., 2018)	99.43	90.57	84
Proposed VGG16-50%	98.35	89.33	82.83
Proposed ResNet50-50%	99.50	95.05	88.73

Performance comparison of the pruned model against the state-of-the-art methods is summarized in Table 4.2. For this comparison, a desiccated model with pruning rate 50% is selected as it is retaining the original performance and still not showing any meaningful performance drop in most of the benchmark. The LBP models are non-deep learning approaches while CenterFace, SphereFace and VGGFace2 are deep-learning based models. It is clearly shown that the deep learning models outperform the conventional machine learning models by a large margin. Furthermore, the proposed method shows comparable performance with the other well-known deep learning model. However, it should be noted that the proposed models are relatively smaller and more efficient. As a fair comparison, the proposed model with VGG16 as the backbone is based on the same model with VGGFace2. However, even though 50% of the parameters in the proposed model are removed, it achieves better performance than the VGGFace2 model.

Chapter 5

Real-time Face Identification System

A complete face identification system consists of detection and recognition modules. This section explains the attempt to combines these modules into a working system. It uses the fast face detector explained in Chapter 3 and the face embedding extractor described in Chapter 4.

The system is deployed on a personal computer with a 3.4GHz Intel i7-4770 processor and 16GB RAM running Ubuntu 18.04.3 LTS operating system. To achieve a decent acceleration on the deep learning model, the experiments utilize a GTX 1060 GPU with 6GB RAM with CUDA 10 driver. All modules are coded in TensorFlow 1.14.

The system employs an RGB Camera with USB connector. This camera works at 30 fps and the resolution is set at 640x480 pixels (VGA). The image streams from this camera are fed into the face detection module with OpenCV as its reader.

To assess the efficiency of the proposed system, the experiments evaluate several setups. The MTCNN model is set as a baseline as it is a quite popular deep learning-based face detection model. It has 0.691, 0.664, and 0.424 of mAP in the WIDER face benchmark on easy, medium, and hard validation sets, respectively.

The basic model in the proposed system is consists of the S3FD model as the face detector. Faces ROIs are aligned using the ONet from MTCNN (3rd stage of the MTCNN) and then fed into the face identification module. The first nearest neighbor is set as the final prediction for face identification. More neighbor can be used for better performance but a single sample is sufficient for small scale identification

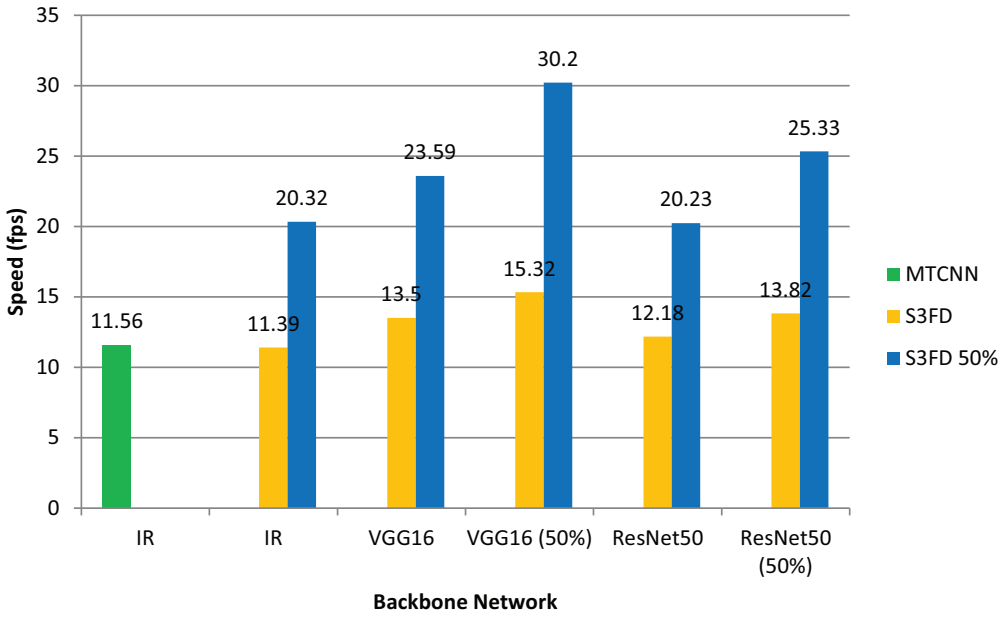

FIGURE 5.1: Processing speed comparison accross different setups (GTX1060Ti).

scenarios.

Figure 5.1 shows the comparison of the processing speed between a complete system which employs MTCNN and S3FD as its face detector. Both of the systems run at a similar speed (11.56 fps vs 11.39 fps) but it should be noted that the S3FD model is far more accurate as it achieves 93.7, 92.5, and 85.1 mAP in the WIDER face detection benchmark for easy, medium, and hard validation sets, respectively. Both of the models use Facenet with Inception-Resnet backbone as an embedding extractor for the identification. For this Facenet model, an opensource implementation (DavidSandberg, 2019) is used.

The experiment then examines the effect of channel pruning on the overall processing speed. When the face detector is pruned with a 50% ratio, it increases the overall processing speed to 20.32 fps, almost 2 times faster compared to the original model. It should be noted that the processing speed of face alignment itself is almost negligible (1ms) and the remaining module with heavy computation is the embedding extractor network.

To improve the processing speed of the identification module, a face identification module based-on VGG16 backbone is utilized. The overall processing speed is slightly improved from 11.39 fps to 13.5 fps due to the smaller size of the backbone.

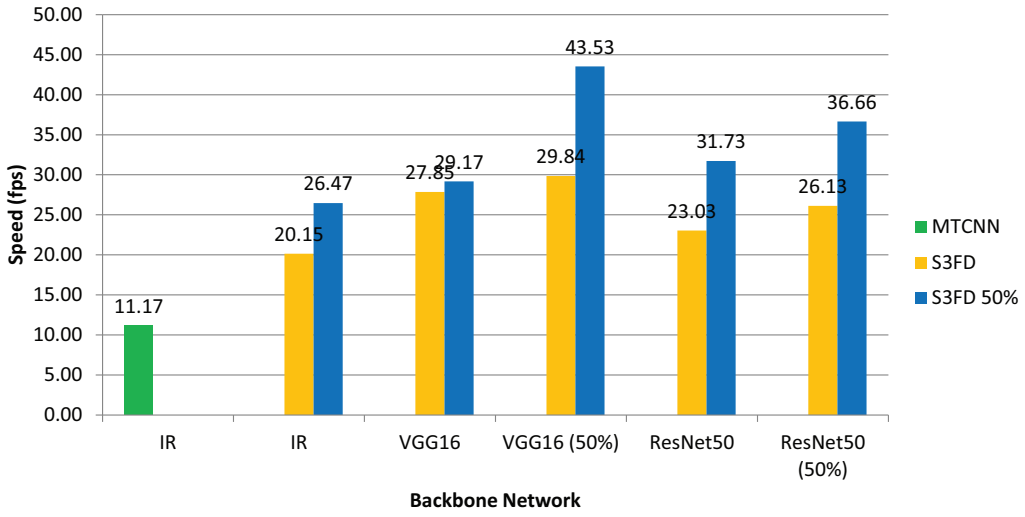

FIGURE 5.2: Processing speed comparison accross different setups (GTX1080Ti).

However, this VGG16-based model is 2% lower accuracy compared to the inception resnet model in terms of validation accuracy in the LFW benchmark.

As explained in Chapter 3, the processing speed of a CNN model is improved through model pruning. With a combination of pruned S3FD and non-pruned VGG16 embedding extractor, the overall processing speed reaching 23.59 fps. This speed approximately consists of 25ms detection, 1ms of alignment and 15 ms of identification.

Pruning on both face detector and identification network gives satisfying speed improvement. With pruning only on the identification module, the speed is improved from 13.5fps to 15.32 fps. If both of the networks are pruned, then the overall speed is reaching a real-time processing speed, 30.20 fps.

From the experiments, it is shown that the key to speed improvement lies in the face detector. It is because most of the processing time is spent by the face detector since it processes a large image compared to the other modules. Furthermore, the current implementation is considering 6 scales of detection where the first scale (corresponds to the smallest scale) accounts for 75% of the total computations.

Several experiments carried out on a different platform is also performed. The system is deployed on a personal computer with a 3.6GHz AMD Ryzen 7-1800X processor and 32GB RAM running Ubuntu 16.04 LTS operating system. To achieve a decent acceleration on the deep learning model, the experiments utilize a GTX 1080Ti GPU with 11GB RAM and CUDA 10 drivers. The system runs at faster computation

speed as shown in the Figure 5.2.

Several examples of the identification output are shown in Figure 5.3. The top image contains 3 faces, one face is partially occluded in the top part, the second face is not occluded, and the third one is painting with an unknown identity. It shows that the proposed systems could reliably identify the faces even with the presence of partial occlusion.

The bottom image shows a similar scenario. In this case, the first face is partially occluded with the presence of a mask. The system still identifies the face properly for the masked face. As indicated in Figure 5.3, the proposed system could not distinguish between real face or painting and even photograph. This problem is not considered in this thesis and left for future work. It may require a higher level of processing and additional sensors such as an infrared camera or depth camera.

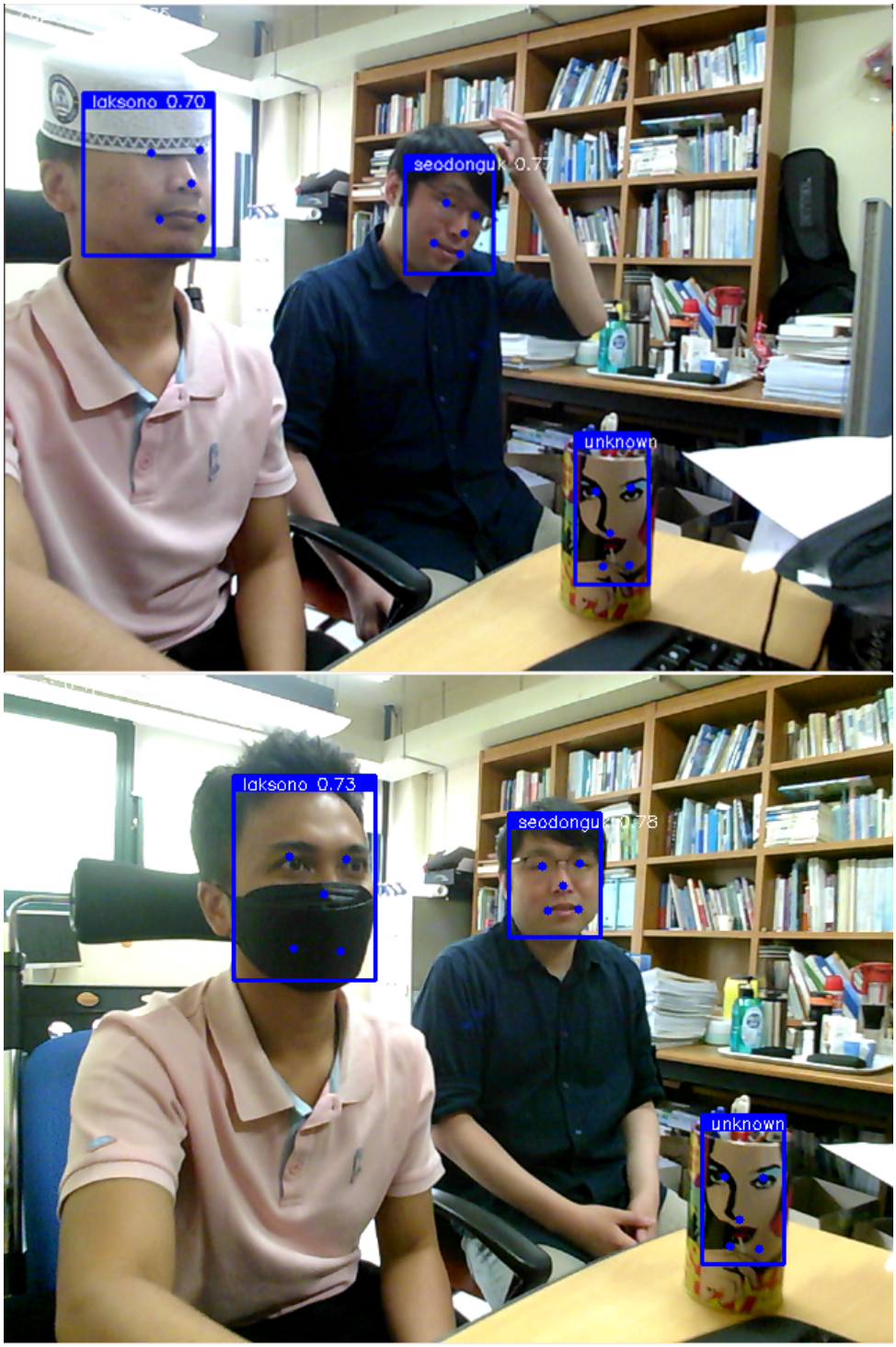

FIGURE 5.3: Examples of the face identification result.

Chapter 6

Conclusion

The works in this thesis focus on a complete face identification system which consists of efficient methods on face detection and recognition networks. A complete pipeline is presented alongside the discussion related to each component.

This thesis examines the utilization of an attention map on the single-shot face detector model. It aims to predict the region around a face that has an impact on the final prediction. According to the experiments, this strategy improves the evaluation score on all test cases within the WIDER evaluation benchmark.

The high-performance face detector is designed based on the single-shot face detector (S3FD) model which predicts the location of faces on several scales. Features pyramid architecture is utilized to enhance the quality of features on the lower-level layer. It combines the information from the lower-level layer with the features from the higher-level layer that has richer semantic information. Therefore, together with the attention map, it delivers excellent detection performances.

Processing speed is an important factor in a real-world face identification system. This thesis proposes a pruning strategy to reduce the computation of the face detector while maintaining the detection quality.

The channel-wise pruning strategy is incorporated into the face detector network. It allows the network to works with a lower number of parameters. Specifically, channels in the filter with low importance scores are removed.

To ensure the pruned model has a good detection quality, fine-tuning is performed on the pruned model. Furthermore, an iterative pruning strategy is employed to ensure that the performance of the model is not drastically dropped in each pruning step.

According to the experiments, the iterative pruning strategy allows the model to be pruned until 50% of its original size while maintaining the detection performance. At this pruning rate, the network has 0.5% of mAP reduction on average for all of the validation sets in the WIDER face detection benchmark. At this pruning rate, the face detector works at 52.44 fps for the VGA resolution image which is approximately 1.6 times faster compared to the original model.

For face recognition, this thesis focus on an embedding-based model. A deep learning model is trained to produce discriminative embeddings that close each other whenever the input images are from the same identity and vice versa. The normalized embedding model, i.e. embedding with L_2 norm equal to 1, is chosen as it provides easier comparison by performing dot product between embedding to determines the similarity between them.

This face recognition model is also pruned in order to provide a fast method with comparable performance to the heavy computation model. The same pruning strategy as the face detection model is employed to obtains a compact face recognition model. According to the experiments, the pruned model retains around 98% performance of the original model in most of the pruning steps with 96.8% of accuracy at 81% pruning rate. It means that the model lost 1.7% of accuracy while having 81% fewer parameters.

Studies on the full face identification model are conducted under several scenarios. The baseline is set as an MTCNN face detector and facenet with the Inception-ResNet backbone. This model is one of the popular open-source methods that runs at 11.56 fps, according to the experiments.

The proposed model in this thesis is tested against the baseline model. With the non-pruned face detection model, it runs at a similar speed to the baseline model while outperforming the mAP score by large margins in the WIDER face detection benchmark.

With the proposed model and pruning strategy, it runs in real-time for the whole detection-identification pipeline. This speed is achieved using a low-end consumer GPU GTX 1060 with 6GB RAM. Compared to the baseline, the proposed system runs 2.69 times faster.

According to the experiments, the proposed system provides a reliable model

that could work in real-time and suitable for real-world applications. Several aspects including speed improvement and quality enhancement are left as future works as described in the following section.

6.1 Future Works

In this thesis, the system is focused on models that run on GPU. Currently, research on low-cost devices is getting popular. Therefore, experiments on smaller models that could work on the embedding devices are left as future work.

Another important aspect is the retrieval strategy in the face recognition module. Currently, one nearest neighbor is used in the retrieval strategy. It will be useful to perform experiments with the k-nearest neighbor model. Furthermore, faster retrieval methods such as the hash-based method or hashed embedding vector should also be examined to determine their effectiveness in improving the processing speed of a face identification system.

Appendix A

Publications

A.1 Journal

1. Laksono Kurnianggoro, Wahyono, and Kang-Hyun Jo, A Survey of 2D Shape Representation: Methods, Evaluations, and Future Research Directions, Neurocomputing, 2018.
2. Yang Yu, Laksono Kurnianggoro, and Kang-Hyun Jo, Moving Object Detection for a Moving Camera Based on Global Motion Compensation and Adaptive Background Model, International Journal of Control, Automation, and Systems, pp.8, 2018.
3. Danilo Caceres Hernandez, Laksono Kurnianggoro, Alexander Filonenko, and Kang-Hyun Jo, Real-Time Lane Region Detection Using Combination of Geometrical and Image Features, Sensors, 2016.
4. Laksono Kurnianggoro, Van-Dung Hoang, and Kang-Hyun Jo, Calibration of a 2D Laser Scanner System and Rotating Platform using a Point-Plane Constraint, Computer Science and Information Systems, Vol.12, No.1, pp.307-322, January 2015.

A.2 Conference

1. Laksono Kurnianggoro and Kang-Hyun Jo, Attention-Guided Model for Robust Face Detection System, PSIVT, Sydney, Australia, Nov 18, 2019.
2. Laksono Kurnianggoro and Kang-Hyun Jo, Optimized Latent Features for Deep Image Compression, HSI2019, Richmond, USA, Jun 25, 2019.

3. Laksono Kurnianggoro and Kang-Hyun Jo, Ensemble of Predictions from Augmented Input as Adversarial Defense for Face Verification System, ACIIDS 2019, Yogyakarta, Indonesia, Apr 8, 2019.

4. Ashraf Uddin Russo, Laksono Kurnianggoro, and Kang-Hyun Jo, Classification of sports videos with combination of deep learning models and transfer learning, ECCE 2019, Coxs Bazar, Bangladesh, Feb 7, 2019 pp.5.

5. Laksono Kurnianggoro and Kang-Hyun Jo, Towards an Integrated Method of Detection and Description for Face Authentication System, HSI 2018, Gdansk, Poland, Jul 4, 2018.

6. Yang Yu, Laksono Kurnianggoro, and Kang-Hyun Jo, Vehicle Contour Segmentation from 3D Point Cloud, CICIRO2017, Changwon, Korea, Nov 30, 2017 pp.3.

7. Laksono Kurnianggoro and Kang-Hyun Jo, Identification of Pedestrian Attributes using Deep Networks, IECON 2017, Beijing, China, Oct 29, 2017.

8. Tang Qing, Laksono Kurnianggoro, Kang-Hyun Jo, Traffic Sign Classification with Dataset Augmentation and Convolutional Neural Network, ICGIP 2017, Qingdao, China, Oct 14, 2017.

9. Laksono Kurnianggoro, Wahyono, Alexander Filonenko, and Kang-Hyun Jo, Shape Classification using Combined Features, ICCCI 2017, Cyprus, Sep 27, 2017.

10. Alexander Filonenko, Laksono Kurnianggoro, and Kang-Hyun Jo, Smoke Detection on Video Sequences using Convolutional and Recurrent Neural Network, ICCCI 2017, Cyprus, Sep 27, 2017.

11. Alexander Filonenko, Laksono Kurnianggoro, and Kang-Hyun Jo, Comparative Study of Modern Convolutional Neural Networks for Smoke Detection on Image Data, HSI 2017, Ulsan, Korea, Jul 17, 2017.

12. Laksono Kurnianggoro and Kang-Hyun Jo, Object Classification for LIDAR Data using Encoded Features, HSI 2017, Ulsan, Korea, Jul 17, 2017.

13. Ajmal Shahbaz, Laksono Kurnianggoro, Wahyono, and Kang-Hyun Jo, Recent Advances in the Field of Foreground Detection: An Overview, ACIIDS 2017, Kanazawa, Japan, Apr 3, 2017.

14. Laksono Kurnianggoro, Wahyono, and Kang-Hyun Jo, Improving Traffic Sign Recognition Using Low Dimensional Features, ACIIDS 2017, Kanazawa, Japan, Apr 3, 2017.

15. Wahyono, Alexander Filonenko, Laksono Kurnianggoro, and Kang-Hyun Jo, A Fuzzy Model-based Integration Framework for Vision-based Intelligent Surveillance Systems, IEEE-ICM 2017, Gippsland, Australia, Feb 14, 2017 pp.358-361.

16. Tang Qing, Laksono Kurnianggoro, and Kang-Hyun Jo, Statistical and Geometrical Features for LiDAR-based Vehicle Detection, SII 2016, Sapporo, Japan, Dec 13, 2016.

17. Laksono Kurnianggoro, Dongwook Seo, Joko Hariyono, Ajmal Shahbaz, and Kang-Hyun Jo, Coarse-to-fine Approach for Fast Correlation-based Visual Tracking, IECON 2016, Florence, Italy, Oct 23, 2016.

18. Ajmal Shahbaz, Laksono Kurnianggoro, Kang-Hyun Jo, Parameter Analysis of Probabilistic Foreground Detector, IECON 2016, Florence, Italy, Oct 23, 2016.

19. Joko Hariyono, Ajmal Shahbaz, Laksono Kurnianggoro and Kang-Hyun Jo, Estimation of Collision Risk for Improving Driver"s Safety, IECON 2016, Florence, Italy, Oct 23, 2016.

20. Laksono Kurnianggoro, Ajmal Shahbaz, and Kang-Hyun Jo, Dense Optical Flow in Stabilized Scenes for Moving Object Detection from a Moving Camera, ICCAS 2016, Gyeongju, Korea, Oct 16, 2016.

21. Ajmal Shahbaz, Laksono Kurnianggoro, Kang-Hyun Jo, A Comparative Study for Foreground Detection using Gaussian Mixture Models-Novice to Novel, ICCAS 2016, Gyeongju, Korea, Oct 16, 2016.

22. Joko Hariyono, Laksono Kurnianggoro, Wahyono and Kang-Hyun Jo, Analysis of Pedestrian Collision Risk using Fuzzy Inference Model, ICCAS 2016, Gyeongju, Korea, Oct 16, 2016.

23. Dongwook Seo, Laksono Kurnianggoro, Joko Hariyono, Danilo Caceres Hernandez, and Kang-Hyun Jo, Compound Road Environment Recognition Method Using Camera and Laser Range Finder, SICE 2016, Tsukuba, Japan, Sep 20, 2016.

24. Laksono Kurnianggoro, Nguyen Quang Huy, Tang Qing, and Kang-Hyun Jo, Comparative Study of Various Machine Learning Algorithms for Object Recognition on Single Scan 2D LIDAR, ICT-ROBOT 2016 ICT-Robot 2016, Busan, Korea, Sep 7, 2016.

25. Tang Qing, Laksono Kurnianggoro, Nguyen Quang Huy, and Kang-Hyun Jo, Vehicle Detection Using LiDAR in Real Road Based on Artificial Neural Network, ICT-ROBOT 2016, Busan, Korea, Sep 7, 2016.

26. Wahyono, Laksono Kurnianggoro, Yang Yu, and Kang-Hyun Jo, A Similarity-based Approach for Shape Classification Using Region Decomposition, ICIC 2016 LNCS 9772, Lanzhou,China, Aug 2, 2016 pp.279-289.

27. Yang Yu, Laksono Kurnianggoro, Wahyono, and Kang-Hyun Jo, Online Programming Design of Distributed System based on Multi-level Storage, ICIC 2016, Lanzhou,China, Aug 2, 2016 pp.10.

28. Laksono Kurnianggoro, Wahyono, Yang Yu, Danilo Caceres Hernandez, and Kang-Hyun Jo, Online Background Subtractor with Motion Compensation for Freely Moving Camera, ICIC 2016, Lanzhou,China, Aug 2, 2016.

29. Laksono Kurnianggoro, Joko Hariyono, Hyun-Deok Kang, and Kang-Hyun Jo, Gaussian-Polynomial Kernel for Real-time Tracking by Detection Method, FCV 2016, Takayama, Japan, Feb 17, 2016.

30. Laksono Kurnianggoro and Kang-Hyun Jo, Real-time Car Detection using Calibrated Camera and Laser Range Finder, ICCAS 2015, Busan, Korea, Oct 13, 2015.

31. Yang Yu, Laksono Kurnianggoro, and Kang-Hyun Jo, The Car Washing Control Method using 3D Contour Segmentation, ICCAS 2015, Busan, Korea, Oct 13, 2015 pp.3.

32. Wahyono, Laksono Kurnianggoro, and Kang-Hyun Jo, Traffic Sign Recognition and Tracking for a Vision-based Autonomous Vehicle Using Optimally Selected Features, SICE 2015, Hangzhou, China, Jul 28, 2015 pp.1419-1422.

33. Yang Yu, Laksono Kurnianggoro, and Kang-Hyun Jo, Design of Intelligent Car Washing System, SICE 2015, Hangzhou, China, Jul 28, 2015 pp.4.

34. Danilo Caceres Hernandez, Alexander Filonenko, Laksono Kurnianggoro, Dongwook Seo, and Kang-Hyun Jo, Iterative Road Detection Based Vehicle Speed , HSI2015 8th International Conference on Human System Iteraction, Warsaw, Poland, Jun 25, 2015.

35. Alexander Filonenko, Danilo Caceres Hernandez, Laksono Kurnianggoro, Dongwook Seo, and Kang-Hyun Jo, Real-Time Lane Marking Detection, CYBCONF 2015, Gdynia, Poland, Jun 24, 2015.

36. Yang Yu, Laksono Kurnianggoro, and Kang-Hyun Jo, Design of Automatic Water Cannon System based on 2D Laser Scanner, ICROS 2015, Daejon, Korea, May 6, 2015 pp.2.

37. Laksono Kurnianggoro, Yang Yu, and Kang-Hyun Jo, Synchronization of a 2D Laser Scanner and Rotating Platform for Wide Range 3D Measurement, ICROS 2015, Daejon, Korea, May 6, 2015.

38. Laksono Kurnianggoro, Wahyono, and Kang-Hyun Jo, Utilization of Optimally Selected Features for Car Detection in Calibrated Camera and LRF System, FCV 2015, Mokpo, Korea, Jan 28, 2015.

39. Wahyono, Van-Dung Hoang, Laksono Kurnianggoro, and Kang-Hyun Jo, Scalable Histogram of Oriented Gradients for Multi-size Car Detection, MECATRONICS 2014, Tokyo, Japan, Nov 27, 2014.

40. Laksono Kurnianggoro, Wahyono, and Kang-Hyun Jo, Real-time Car Detection and Tracking using Calibrated System of Camera and LRF, Korean Multimedia Society (KMS) 2014 Korean Multimedia Society, Gyeongju, Korea, Nov 14, 2014 pp.107-108.

41. Van-Dung Hoang, Dongwook Seo, Laksono Kurnianggoro, and Kang-Hyun Jo, Path planning and global trajectory tracking control assistance to autonomous vehicle, URAI 2014, Kuala Lumpur, Malaysia, Nov 12, 2014.

42. Wahyono, Laksono Kurnianggoro, Dongwook Seo, and Kang-Hyun Jo, Visual Perception of Traffic Sign for Autonomous Vehicle Using k-Nearest Cluster Neighbor Classifier, URAI 2014, Kuala Lumpur, Malaysia, Nov 12, 2014.

43. Wahyono, Laksono Kurnianggoro, Joko Hariyono, and Kang-Hyun Jo, Traffic Sign Recognition System for Autonomous Vehicle Using Cascade SVM Classifier, IECON 2014, Texas, USA, Oct 29, 2014.

44. Laksono Kurnianggoro, Wahyono, Danilo Caceres Hernandez, and Kang-Hyun Jo, Camera and Laser Range Finder Fusion for Real-time Car Detection, IECON 2014 Industrial Electronic Conference 2014, Texas, USA, Oct 29, 2014 pp.3419-3424.

45. Laksono Kurnianggoro, Van-Dung Hoang, and Kang-Hyun Jo, Calibration of a Rotating 2D Laser Range Finder using Circular Path on a Plane Constraints, ICCCI 2014 Studies in Computational Intelligence - New Trends in Computational Collective Intelligence, Seoul, Korea, Sep 24, 2014 pp.155-163.

46. Wahyono, Laksono Kurnianggoro, Joko Hariyono, and Kang-Hyun Jo, Similarity-Based Classification of 2-D Shape Using Centroid-Based Tree-Structured Descriptor, IEA-AIE 2014, Kaohsiung, Taiwan, Jun 3, 2014.

47. Joko Hariyono, Laksono Kurnianggoro, Wahyono, Danilo Caceres Hernandez, and Kang-Hyun Jo, Ego-motion Compensated for Moving Object Detection in a Mobile Robot, IEA-AIE 2014, Kaohsiung, Taiwan, Jun 3, 2014.

48. Laksono Kurnianggoro and Kang-Hyun Jo, Free Road Space Estimation Based on Surface Normal Analysis in Organized Point Cloud, ICIT 2014, Busan, Korea, Feb 26, 2014.

49. Wahyono, Laksono Kurnianggoro, and Kang-Hyun Jo, A Fast Stroke Width Transform Image Generating Method for Text Detection, QCAV 2013, Fukuoka, Japan, May 30, 2013.

Bibliography

Bourdev, L. and J. Brandt (2005). "Robust object detection via soft cascade". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2, 236–243 vol. 2. DOI: 10.1109/CVPR.2005.310.

BusinessNewsDaily (2019). *Facial Recognition Advertising: The New Way to Target Ads at Consumers*. URL: https://www.businessnewsdaily.com/15213-walgreens-facial-recognition.html (visited on 10/07/2019).

Cai, Zhaowei et al. (2016). "A unified multi-scale deep convolutional neural network for fast object detection". In: *european conference on computer vision*. Springer, pp. 354–370.

Cao, Qiong et al. (2018). "Vggface2: A dataset for recognising faces across pose and age". In: *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, pp. 67–74.

Chi, Cheng et al. (2019). "Selective refinement network for high performance face detection". In: *Association for the Advancement of Artificial Intelligence (AAAI)*.

Chopra, Sumit, Raia Hadsell, Yann LeCun, et al. (2005). "Learning a similarity metric discriminatively, with application to face verification". In: *CVPR (1)*, pp. 539–546.

Comparitech (2019). *The world's most-surveilled cities*. URL: https://www.comparitech.com/vpn-privacy/the-worlds-most-surveilled-cities/ (visited on 10/07/2019).

Cortes, Corinna and Vladimir Vapnik (1995). "Support vector machine". In: *Machine learning* 20.3, pp. 273–297.

Dai, Jifeng et al. (2016). "R-fcn: Object detection via region-based fully convolutional networks". In: *Advances in neural information processing systems*, pp. 379–387.

Dalal, N. and B. Triggs (2005). "Histograms of oriented gradients for human detection". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

DavidSandberg (2019). *Face recognition using Tensorflow*. URL: https://github.com/davidsandberg/facenet/ (visited on 10/31/2019).

Davis, Jason V et al. (2007). "Information-theoretic metric learning". In: *Proceedings of the 24th international conference on Machine learning*. ACM, pp. 209–216.

Deng, Jiankang et al. (2019a). "Arcface: Additive angular margin loss for deep face recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699.

Deng, Jiankang et al. (2019b). "RetinaFace: Single-stage Dense Face Localisation in the Wild". In: *arXiv preprint arXiv:1905.00641*.

Dong, Xuanyi and Yi Yang (2019). "Searching for a robust neural architecture in four GPU hours". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1761–1770.

Felzenszwalb, Pedro F et al. (2009). "Object detection with discriminatively trained part-based models". In: *IEEE transactions on pattern analysis and machine intelligence* 32.9, pp. 1627–1645.

Frankle, Jonathan and Michael Carbin (2019). "The lottery ticket hypothesis: Finding sparse, trainable neural networks". In: *Advances in neural information processing systems*.

Freund, Yoav, Robert Schapire, and Naoki Abe (1999). "A short introduction to boosting". In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780, p. 1612.

Freund, Yoav and Robert E Schapire (1996). "Schapire R: Experiments with a new boosting algorithm". In: *Thirteenth International Conference on ML*. Citeseer.

Fukushima, Kunihiko (1979). "Neural network model for a mechanism of pattern recognition unaffected by shift in position-Neocognitron". In: *IEICE Technical Report, A* 62.10, pp. 658–665.

Girshick, Ross (2015). "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448.

Girshick, Ross et al. (2014). "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.

Han, Song, Huizi Mao, and William J Dally (2016). "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding". In: *Advances in neural information processing systems*.

Han, Song et al. (2015). "Learning both weights and connections for efficient neural network". In: *Advances in neural information processing systems*, pp. 1135–1143.

He, K. et al. (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

Huang, Gary B. et al. (2007). *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. Tech. rep. 07-49. University of Massachusetts, Amherst.

Kemelmacher-Shlizerman, Ira et al. (2016). "The megaface benchmark: 1 million faces for recognition at scale". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4873–4882.

Koestinger, Martin et al. (2012). "Large scale metric learning from equivalence constraints". In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE, pp. 2288–2295.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1097–1105.

Lecun, Y. et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324. DOI: 10.1109/5.726791.

Li, Jian et al. (2019). "DSFD: Dual Shot Face Detector". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Liang, Jason, Elliot Meyerson, and Risto Miikkulainen (2018). "Evolutionary architecture search for deep multitask networks". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, pp. 466–473.

Lin, Tsung-Yi et al. (2017). "Feature pyramid networks for object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125.

Liu, Chenxi et al. (2018a). "Progressive neural architecture search". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 19–34.

Liu, Hanxiao, Karen Simonyan, and Yiming Yang (2019). "Darts: Differentiable architecture search". In: *Advances in neural information processing systems*.

Liu, Wei et al. (2016). "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer, pp. 21–37.

Liu, Weiyang et al. (2017). "Sphereface: Deep hypersphere embedding for face recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 212–220.

Liu, Weiyang et al. (2018b). "Learning towards minimum hyperspherical energy". In: *Advances in Neural Information Processing Systems*, pp. 6222–6233.

Liu, Yuanliu et al. (2018c). "iqiyi-vid: A large dataset for multi-modal person identification". In: *arXiv preprint arXiv:1811.07548*.

Mathias, Markus et al. (2014). "Face detection without bells and whistles". In: *European conference on computer vision*. Springer, pp. 720–735.

Moschoglou, Stylianos et al. (2017). "Agedb: the first manually collected, in-the-wild age database". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 51–59.

Najibi, Mahyar et al. (2017). "Ssh: Single stage headless face detector". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4875–4884.

Parkhi, O. M., A. Vedaldi, and A. Zisserman (2015). "Deep Face Recognition". In: *British Machine Vision Conference*.

Pham, Hieu et al. (2018). "Efficient neural architecture search via parameter sharing". In: *arXiv preprint arXiv:1802.03268*.

Redmon, Joseph and Ali Farhadi (2017). "YOLO9000: better, faster, stronger". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271.

Redmon, Joseph et al. (2016). "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.

Ren, Shaoqing et al. (2015). "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*, pp. 91–99.

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

Sengupta, Soumyadip et al. (2016). "Frontal to profile face verification in the wild". In: *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 1–9.

Simonyan, K., A. Vedaldi, and A. Zisserman (2014). "Learning Local Feature Descriptors Using Convex Optimisation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.8, pp. 1573–1585. DOI: 10.1109/TPAMI.2014.2301163.

SouthChinaMorningPost (2019a). *As China turns towards facial recognition payments, are QR codes on their way out?* URL: https://www.scmp.com/tech/enterprises/article/3020657/china-turns-towards-facial-recognition-payments-are-qr-codes-their (visited on 10/07/2019).

— (2019b). *DeepGlint: the Chinese AI firm that helped police catch a criminal who had been on the run for 20 years*. URL: https://www.scmp.com/tech/start-ups/article/3008998/deepglint-chinese-ai-firm-helped-police-catch-criminal-who-had-been (visited on 10/07/2019).

— (2019c). *Shenzhen AI start-up Intellifusion helps city police identify jaywalkers and banned drivers*. URL: https://www.scmp.com/tech/start-ups/article/3008700/shenzhen-ai-start-intellifusion-helps-city-police-identify (visited on 10/07/2019).

Szegedy, Christian et al. (2016). "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826.

Szegedy, Christian et al. (2017). "Inception-v4, inception-resnet and the impact of residual connections on learning". In: *Thirty-First AAAI Conference on Artificial Intelligence*.

Tang, Xu et al. (2018). "Pyramidbox: A context-assisted single shot face detector". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 797–813.

TargetMarketing (2019). *Facial Recognition: Ads Target Consumers for You*. URL: https://www.targetmarketingmag.com/article/facial-recognition-ads-target-consumers/all (visited on 10/07/2019).

Uijlings, Jasper RR et al. (2013). "Selective search for object recognition". In: *International journal of computer vision* 104.2, pp. 154–171.

Viola, Paul, Michael Jones, et al. (2001). "Rapid object detection using a boosted cascade of simple features". In: *CVPR (1)* 1.511-518, p. 3.

Wahyono, Joko Hariyono, and Kang-Hyun Jo (2017). "Body part boosting model for carried baggage detection and classification". In: *Neurocomputing* 228. Advanced Intelligent Computing: Theory and Applications, pp. 106 –118. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2016.10.038. URL: http://www.sciencedirect.com/science/article/pii/S0925231216312589.

Wahyono and K. Jo (2017). "Cumulative Dual Foreground Differences for Illegally Parked Vehicles Detection". In: *IEEE Transactions on Industrial Informatics* 13.5, pp. 2464–2473. DOI: 10.1109/TII.2017.2665584.

Wang, Feng et al. (2017). "Normface: l2 hypersphere embedding for face verification". In: *Proceedings of the 25th ACM international conference on Multimedia*. ACM, pp. 1041–1049.

Wang, Hao et al. (2018). "Cosface: Large margin cosine loss for deep face recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5265–5274.

Wang, Jianfeng, Ye Yuan, and Gang Yu (2017). "Face attention network: An effective face detector for the occluded faces". In: *arXiv preprint arXiv:1711.07246*.

Wen, Yandong et al. (2016). "A discriminative feature learning approach for deep face recognition". In: *European conference on computer vision*. Springer, pp. 499–515.

Whitelam, Cameron et al. (2017). "Iarpa janus benchmark-b face dataset". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 90–98.

Wolf, Lior, Tal Hassner, and Itay Maoz (2011). *Face recognition in unconstrained videos with matched background similarity*. IEEE.

Yang, Bin et al. (2014). "Aggregate channel features for multi-view face detection". In: *IEEE international joint conference on biometrics*. IEEE, pp. 1–8.

Yang, Shuo et al. (2016). "WIDER FACE: A Face Detection Benchmark". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yi, Dong et al. (2014). "Learning face representation from scratch". In: *arXiv preprint arXiv:1411.7923*.

Zhang, Shifeng et al. (2017). "S3fd: Single shot scale-invariant face detector". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 192–201.

Zheng, Tianyue and Weihong Deng (2018). "Cross-pose lfw: A database for studying crosspose face recognition in unconstrained environments". In: *Beijing University of Posts and Telecommunications, Tech. Rep*, pp. 18–01.

Zheng, Tianyue, Weihong Deng, and Jiani Hu (2017). "Cross-age lfw: A database for studying cross-age face recognition in unconstrained environments". In: *arXiv preprint arXiv:1708.08197*.

Zhu, Chenchen et al. (2018). "Seeing Small Faces from Robust Anchor's Perspective". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5127–5136.

Zoph, Barret et al. (2018). "Learning transferable architectures for scalable image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710.