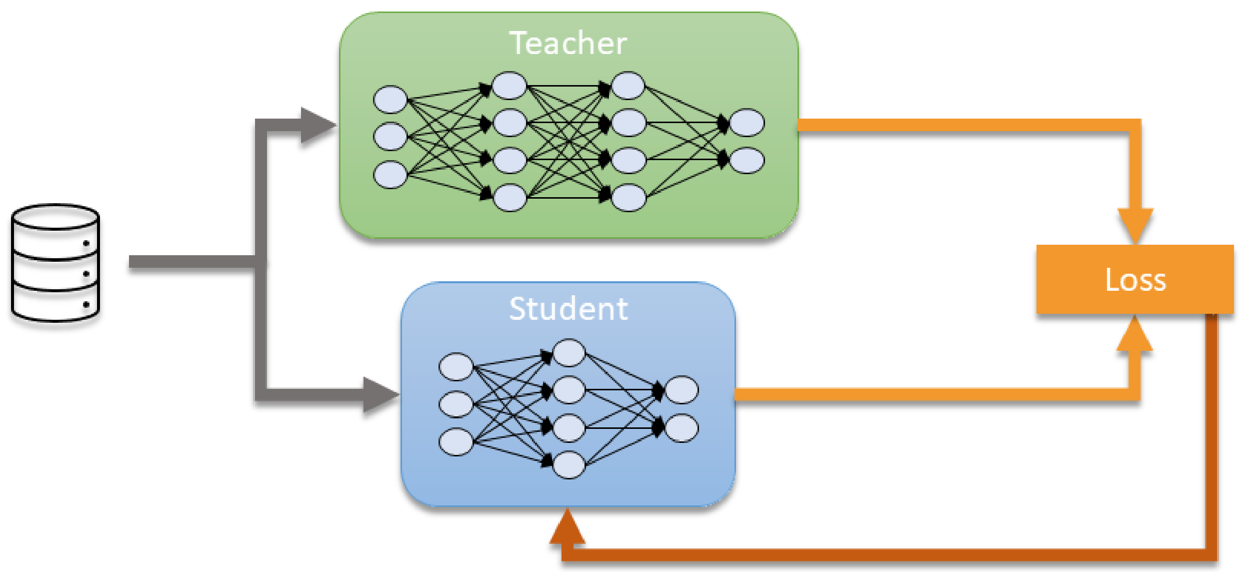
CHALMERS
UNIVERSITY OF TECHNOLOGY



Knowledge Distillation for Face Recognition on Lightweight Neural Networks

Master's thesis in Master Programs Biomedical Engineering and System, Control and Mechatronics

LOUISE GERTZ & ALBIN LINDMARK

Department Of Electrical Engineering
Chalmers University Of Technology
Gothenburg, Sweden 2021

Knowledge Distillation for Face Recognition on Lightweight Neural Networks

Louise Gertz

Albin Lindmark

Department of Electrical Engineering
Division of Signals and Systems
Chalmers University of Technology
Gothenburg, Sweden 2021

Knowledge Distillation for Face Recognition on Lightweight Neural Networks
Louise Gertz
Albin Lindmark

Supervisor: Kenneth Jonsson, Smart Eye
Supervisor: Ahmet Oguz Kislal, Department of Electrical Engineering
Examiner: Marija Furdek Prekratic, Department of Electrical Engineering

EENX30
Department of Electrical engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Printed by Department of Electrical Engineering
Gothenburg, Sweden 2021

Abstract

Face recognition is a common biometric used in everyday commercial products and is also widely used in safety and surveillance. Accuracy is critical when face recognition is used for authentication purposes. Implementation of accurate face recognition using Convolutional Neural Network (CNN) models is limited to deployment in high-end complex systems due to the computational complexity. Viable implementations of accurate face recognition in mobile devices demands less computationally expensive methods, such as smaller models. This thesis investigates the potential of knowledge distillation (KD), a machine learning technique used to improve the performance of a small model by transferring knowledge from a larger model to the small one. KD was implemented on CNNs trained for the task of face identification and verification using low resolution near infrared images. Both the identification and verification models trained with KD achieved a higher accuracy than the reference models trained with standard procedures. Methods using a staged training procedure or hints comparing features of the models were shown to further improve KD and to be useful when there is a large discrepancy between model sizes. Training using KD was proven to increase learning, thus making it possible to increase the accuracy of small face recognition networks.

Keywords: knowledge distillation, machine learning, convolutional neural networks, lightweight neural networks, face recognition, face identification, face verification, near infrared images, artificial intelligence

Acknowledgements

We would like to express our gratitude to Smart Eye AB for giving us the opportunity to work on this interesting project. Many thanks to Kenneth Jonsson for sharing his knowledge and providing us with technical guidance and support throughout the project.

Further, we want to thank our examiner Marija Furdek Prekratic, at the Department of Electrical Engineering, for taking on this thesis project and for offering us valuable feedback. A special thanks to Oguz Kislal, at the Department of Electrical Engineering, for invaluable advice, support and continuous feedback on the report.

Lastly, we would like to express our sincere gratitude to everyone who has helped us during this project and for making it a memorable journey.

Louise and Albin, January 2021

Contents

List of Figures

List of Tables

1 Introduction

During the last decade artificial intelligence (AI) research and use of AI features increased extensively. According to the AI Index Report by Shoham, the growth of AI startups in the U.S. was 113 % between 2015-2018 [1]. One AI domain rapidly gaining popularity is computer vision, with functionalities such as face detection, face recognition and facial expression recognition. Face recognition employs deep neural networks with the goal of recognizing the face of the input image from a data set of images. The use of face recognition can be found in everyday technology such as cellphones but is also widely used in applications in law enforcement, surveillance, marketing and biometrics [2].

Smart Eye AB is a company developing eye-tracking technology that aims to understand, assist and predict human intentions and actions. The company provides face recognition as a part of its offerings in the automotive market segment. The majority of today's automobiles are not equipped with a graphics processing unit (GPU) which limits their computational power, making it difficult to implement face recognition [3]. Face recognition could potentially be used as an identification method for secure payments in the automotive industry. This raises the requirements of a low false acceptance rate (FAR) as well as a low false rejection rate (FRR). The models currently employed by Smart Eye are low complexity models which allow execution on ARM-based devices but do not meet said requirements.

One way of addressing this problem is known as knowledge distillation (KD), a model compression method where knowledge is transferred from a large model (the teacher) to a small model (the student). One of the benefits of KD is that the student often does not have to absorb all the original raw data and can therefore be several orders of magnitude smaller than the teacher, yet often performs on a level comparable to the teacher. To achieve this, most KD-algorithms use a teacher model which is trained on the entirety of the data, just as in normal learning. To train the student a special loss-function that compares the output from the teacher and student is used to improve the efficiency of the students learning. In the case of face identification using image data, a type of model known as Convolutional Neural Network (CNN) is commonly used. CNNs are good at reducing the complexity of high dimensional inputs such as images and can be used both as teacher and as student. A great example of a successful implementation of KD on a CNN architecture is VarGFaceNet, one winner of the “Lightweight Face Recognition Challenge” held in Seoul, Korea, 2019 [2].

1.1 Aim

The foundation of this research surrounds the recurring problem that machine learning face recognition implementations with a high accuracy are generally too computationally complex for most commercial products. Most embedded systems of today can only handle CNN models of low computational complexity which are not suitable for high accuracy applications because of their poor performance.

Knowledge distillation has been shown to be able to improve the training of the network by transferring knowledge from a larger teacher model to a smaller student model. This thesis intends to find out how knowledge distillation can improve the performance of small CNNs of around 100K parameters. The demarcation of the model size is to ensure suitability for the automotive industry where computational power is limited.

Furthermore, the research investigates if the performance gains of knowledge distillation are applicable in the face recognition case, investigating both the task of identification and verification, using low quality near infrared (NIR) images. The aim is to determine if knowledge distillation is a promising method which could make face recognition viable in Smart Eye’s embedded low computational cost systems.

1.2 Previous Work

The discipline now known as knowledge distillation was pioneered by methods that approximate assemblies of small classifiers with one large classifier. The idea being to reduce the total number of parameters yet maintaining performance at the level of the original assembly [4] [5]. Following these early techniques, it was shown by Caruana that performance until then achievable only with deep networks was achievable with shallow networks if they were trained to mimic a deep network rather than trained with raw data [6]. This proved that training technique is just as important as architectural complexity. Hinton explored the findings further, showing that knowledge distillation can not only improve the performance of most networks but can for example also reduce the amount of data needed to train a network, prevent over-fitting and teach a finished model additional information and classes [7].

Since then, new techniques to achieve knowledge distillation have been published and many have been successfully applied to a range of problems. Knowledge distillation has also been combined with other techniques such as reinforcement learning and adversarial learning. This has led to a subdividing and categorization of the field depending on the knowledge distilled and the target problem [8]. More importantly, knowledge distillation has remained competitive in the area of image classification. A recent competition “Lightweight Face Recognition Challenge” [9] was won by a model trained using a knowledge distillation approach [10].

Regarding NIR data, a few works have to some extent incorporated NIR information in the KD training process. It has been shown that KD can be used to expand a network trained with three channels of RGB data to use an additional channel of NIR data with a relatively small amount of extra NIR data [11]. KD has also been applied along with specialized loss-functions to identify correlation between RGB and NIR images [12]. However, very little evaluation of KD has been made using only NIR data during training, using very low quality NIR images and using light-weight networks for that purpose.

1.3 Ethics

Face recognition is utilized in many everyday products such as cell phones and websites, for example Facebook. It can be used as a tool to identify an individual but also as a health care tool for a variety of applications, for example monitoring a patient or diagnosing genetic disorders [13]. Face recognition is a useful tool in law enforcement and surveillance where it can be used to verify and identify the identities of suspects and criminals [1].

Investigating the potential of knowledge distillation for face identification and verification could lead to a wider use of these methods, for example in the automotive industry. In the automotive industry it could potentially be used for verification for safe payments and driver verification to ensure that no unauthorized person drives the vehicle. This type of safety features could contribute towards the UN’s Sustainable Development Goals of “Good health and well-being for people” and “Sustainable cities and communities” [14].

Although face identification technology brings positive innovation and development of many health-care and safety features, it is impossible to disregard the growing concerns about privacy and data protection. How will the information be used by police and public security teams and who will own the data? Further, the limited performance of face identification algorithms and bias in the data-sets can lead to misidentification. In the case of law enforcement it could lead to the arrest of innocent people [1].

An ethical framework was released by the Biometrics and Forensics Ethics Group to guide the development of the technology [15]. Responsible ways to use face recognition for public safety in an ethical way are being investigated by governments and companies world-wide. It can be concluded that the development and use of face recognition could have a positive impact on society for example within public safety and health. However, it is important to consider ethics, privacy, human rights and equality during the development of these technologies to ensure that the technology is not used in the wrong way.

In this thesis, images of faces from different angles will be used to train the neural networks. The data base consists of millions of images of thousands of different identities. This data was acquired by Smart Eye and will be used in compliance with General Data Protection Regulation. The data is stored on encrypted servers in accordance to recommendations regarding personal data and is detached from human identities by replacing names by integers. This ensures the privacy of the test subjects taking part in the study.

A majority of the test subjects in the data base are of western European origin which could cause a bias in the data. If the model is trained on mostly white subjects it may have a harder time to recognize subjects of color. To ensure the system predicts just and accurate predictions, regardless of ethnicity, it is important to train with a diverse data base. During principal testing of knowledge distillation only relative improvement of performance is of importance. Therefore, the bias won't affect the results of the thesis. But for this kind of systems to be deployed commercially it is of high importance to eliminate biases of gender or ethnicity.

2

Theory

This study applies advanced machine learning theory such as complex loss functions and training algorithms. These concepts are explained in detail in the following chapter, but to further enable the reader to understand them, a review of basic machine learning theory is also conducted. This includes an explanation of the basics of artificial neural networks, the concepts of face recognition and a break down of the MobileNetV2 architecture.

2.1 Artificial Neural Networks

The computing system Artificial Neural Networks (ANNs), commonly known as Neural Networks (NN), is a well-known machine learning method inspired by the neural networks of the brain [16]. The networks are constructed in different layers. There is an input layer where data enter, one or more hidden layers and an output layer which produces the predictions of the network, as shown in Figure 1. Each layer consists of neurons, basic computational units which have parameters that are tuned during training. Both the performance and the computational cost of the network generally increase with the number of parameters. The number of neurons and the number of hidden layers determine the width and depth of the model [17].

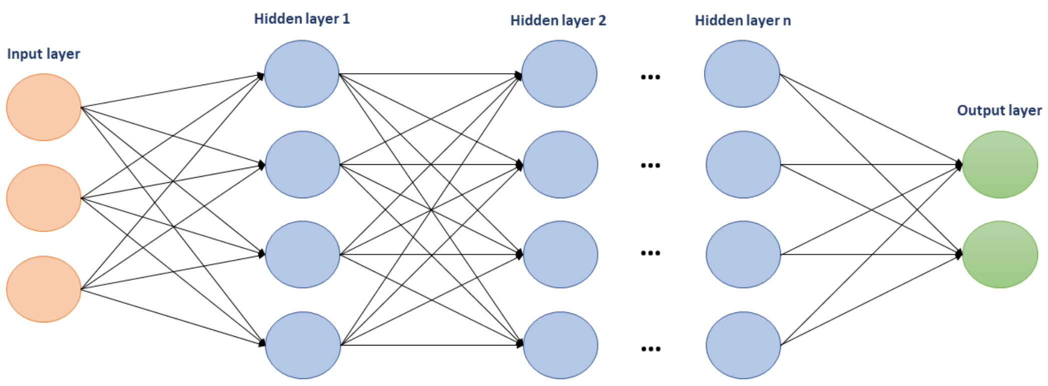


Figure 1: The basic structure of an artificial neural network with n layers.

A neural network can be divided into two parts, the backbone and the top. The backbone is the main part of the network where the the input is transformed into low dimensional representations called features. The top, also known as a classifier, is the final part of the network and is used to correlate features to classes.

A neural network approximates some function f^*, for example a linear classifier $y = f^*(x)$, which maps the input x to the category y [17]. The network defines a mapping of function f, $y = f(x; \Theta)$, and learns the parameters of Θ that produce the best approximation of the function f^*. Loss functions are used to guide the learning procedure in the right direction. A loss function estimates how well the model approximates the relationship between x and y, typically by calculating the difference or distance between the predicted y and the known true y.

The learning procedure can be divided into two steps, forward propagation and back propagation [18]. Forward propagation is the computations performed from the input to the output layer, calculating and storing intermediate variables layer to layer using matrix multiplications. Back propagation on the other hand traverses through the network in reverse. The back propagation algorithm is used to efficiently calculate the gradient of the loss function which is used to update the parameters through gradient descent. The weights and biases are updated for each layer of the network, starting from the output layer propagating backwards.

2.1.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a type of NNs commonly used for computer vision tasks [19]. What sets CNNs apart from the typical NNs is the use of convolution, in at least one layer, instead of the general matrix multiplication [17]. The convolutional layer performs multiplication of the two-dimensional input array with a filter, a two-dimensional array of learnable weights. The filter has a small receptive field and is moved across the image from top left to bottom right, as described in Figure 2. The dot product of the filter-sized patch of the input and the filter is calculated for each point of the input creating a two-dimensional feature-map.

The task of the filters is to identify the features present in the image, for example edges, lines and bends [17]. During training, the network learns patterns from the images and updates the weights, e.g. filter parameters, in a way that specific features can be detected. Whether or not a feature is present in each location of the image is decided by feeding the feature map through an activation function. The output from the convolutional layer is usually modified with a pooling-layer which performs a down-sampling where the output is replaced with a summary statistics of the outputs in the same neighbourhood. For example a 2×2 max pooling replaces a 2×2 region of the input feature map with one value containing the largest value of said region. If a 2×2 region contains the numbers $[2, 4, 1, 7]$ max pooling will return the value 7 in its place. This reduces the number of parameters, which reduces the computational complexity. The pooling layer is also used to make the feature map locally invariant to translation, meaning that most values of the pooled output should remain the same even when there is a small change of input.

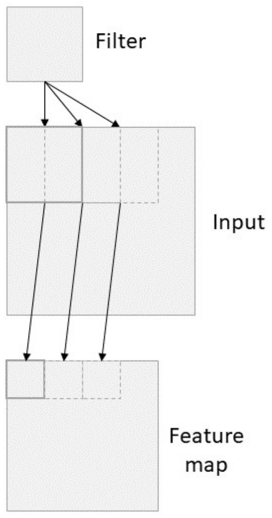


Figure 2: Example of how the receptive field of a filter is moved over the input image to create a feature map.

The concept of sparse interactions and parameter sharing are two motivations for using CNNs [17]. In a traditional NN with m inputs and n outputs then the matrix multiplication calls for $m \times n$ number of parameters and every input unit interacts with every output unit of the layer. A CNN, on the other hand often has sparse interactions since the filter size is smaller than the input size. This results in each neuron only being connected to a limited number of other neurons, which results in a reduction of the number of parameters and operations. Parameter sharing refers to the sharing of the weights between neurons as opposed to traditional NNs where individual elements of the weight matrix are only used once. For a filter in a CNN, one set of weights is learned and reused instead of learning a separate set for every spot in the input. Hence, fewer parameters are used which further improves the computational efficiency of the network.

2.1.2 Face Recognition

Face recognition is a discipline that covers identification and verification of faces. Face identification tries to map images of faces to their corresponding identities, also known as classification, while face verification tries to determine if two images portray the same person [20]. A challenge with large-scale identification and verification is to maintain a distinction between the classes as the number of classes increases.

For faces, it is even more challenging as faces posses large intra-class variability and low inter-class differences. Large intra-class variability can be caused by faces having several expressions each while low inter-class variability is caused by the fact that many human faces naturally look similar to one another [21]. If a network does not achieve high enough inter-class difference and low enough intra-class difference in its feature space, it will fail to distinguish different faces. There are many ways to achieve identification and verification, usually with a trade of between complexity and performance.

2.1.2.1 Softmax with Crossentropy

One way to achieve face recognition is to use a softmax activation on the output of a linear layer. Softmax is a function which can be used to normalize the outputs of a network such that each value in the output lies between 0 and 1 and the sum of all outputs is 1. In this sense, the output from a softmax layer can be regarded as a distribution of pseudo-probabilities where a value of 1 indicates 100% certainty in the network's estimation. This allows easy labeling of data using what is know as a one-hot encoding. The one-hot encoding is a vector with an element for each class where the correct class is indicated using a 1 and the elements for all other classes are set to 0. Using a cross entropy operation between the network's pseudo-probabilities and the one-hot encoding, an estimate of the error in the network's prediction capability can be established. If the output from a softmax function is regarded as an n-dimensional space, then each class is represented with one dimension only and the most prominent dimensional value represents the most plausible class index. This causes the space to grow linearly with the number of classes, as does the number of weights in the linear layer preceding the softmax function. The softmax method does not stimulate the network to separate the classes in the feature space very well. This may cause classes to become impossible to discriminate when the number of classes increases [20].

2.1.2.2 SphereFace and ArcFace

SphereFace introduces a method to overcome many of softmax's weaknesses [20]. SphereFace utilizes the feature encoding (output) of a network as the angular coordinates of a hyper-sphere. Classes are distributed on this hyper-sphere using combinations of the angular coordinates. The softmax is limited to representing one class per dimension but ArcFace can represent multiple classes per dimension because it uses the network output as coordinates instead of indicators, as seen in Figure 3. SphereFace also uses a margin parameter as a penalty to the decision boundaries on this hyper-sphere to force the classes and features to be separated to a greater extent and in a more controlled fashion than with an ordinary softmax.

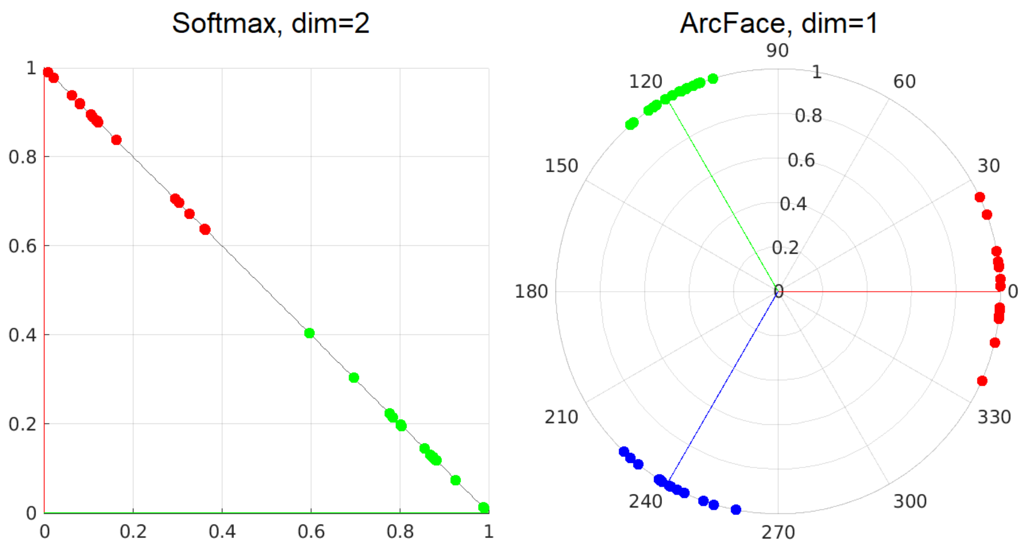


Figure 3: An example of softmax with two dimensions, the x and y axis, and ArcFace with one dimension, the angle. ArcFace can represent more classes with fewer dimensions.

Discrimination between classes is carried out using the cosine-distance to identify which class is most representative of a certain feature encoding. If A is a vector of feature encodings for a data sample and B is a vector representing a class center, then the cosine distance between them is calculated as one minus the dot product over the product of euclidean norms for the two vectors.

$$D = 1 - \frac{A \cdot B}{\|A\|_2 \|B\|_2} \tag{1}$$

The distance from the feature encoding of each data sample to each class center can then be calculated. In this table of distances, the index of the shortest distance for a data sample indicates the most plausible class for that data sample.

Additive Angular Margin Loss (ArcFace) is an improvement to SphereFace, designed to achieve even higher class separability by adjusting the expression for the marginal penalty [22]. This adjustment constructs a linear and constant penalty throughout the hyper-sphere with better class decision boundaries as a result. In addition to greater class separability, SphereFace and ArcFace can also require fewer parameters compared to a softmax configuration when a fully connected classifier is used between the convolutional backbone and the loss. This is because the classifier will not need as many outputs to describe the classes.

The ArcFace loss-function is expressed as

$$L_{arcface} = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{s(cos(\theta_{yi}+m))}}{e^{s(cos(\theta_{yi}+m))} + \sum\limits_{j=1, j\neq y_i}^{n} e^{s(cos(\theta_j))}} \tag{2}$$

where N is batch-size, n is the number of classes, θ_y are the feature encoding and θ_j are the class centers on the sphere. The margin is represented by m and s is a scaling factor.

2.2 Knowledge Distillation

The need for small yet high-performing networks has introduced a number of approaches that try to reduce the computational complexity of networks without a proportional loss in performance. These approaches range from relatively simple methods that reduce the bit depth of the weights, to more complex methods that introduce computational cost as a part of the optimization problem itself [23] [24]. One of the more advanced approaches is known as knowledge distillation and has recently shown capabilities to improve the learning of networks only through better training strategies [23]. Knowledge distillation is able to do this without altering the network's complexity, architecture or training data. Literature shows that networks can achieve almost 175 times fewer parameters with minimally reduced performance [9] as well as networks that achieve higher performance when trained with a KD approach in comparison to a more complex network trained without KD [25].

The area of KD is in itself a multitude of algorithms and techniques, but they all utilize what is known as a teacher-student approach. The basic idea is to deploy a large model (the teacher) to guide a smaller model (the student) while the student is being trained, such that the student is able to absorb more knowledge than during traditional training [7]. In this sense, the knowledge of the teacher is being distilled into the student. The higher concentration of knowledge in the student ideally allows it to perform just as well as the teacher but with less computational resources.

2.2.1 Hinton Algorithm

One of the pioneering knowledge distillation algorithms was developed by Hinton et al. [7] and combines two loss functions in order to achieve distillation. One loss function compares the student's softmax distribution to the one-hot encoding of the data (hard labels) and the other loss function compares the student's and the teacher's softmax distributions (soft labels).

If the teacher model is well taught, it will produce a high pseudo-probability for the correct answer, similar to a one-hot encoding. It will, however, also produce pseudo-probabilities for the classes which are incorrect. These incorrect probabilities may seem like a disadvantage over a true one-hot encoding, but this is not the case. Hinton et al. showed that there is valuable information among the false soft labels that the student can learn from. Imagine teaching a child to identify fruits. When learning about oranges, it may be useful to see how similar the properties of an orange are to those of a grape fruit and also how different they are from those of a pineapple. The loss function for soft labels achieves a very similar effect by showing the student the entire spectrum of class probabilities. Because of this, the student model may also need far less data to absorb all the knowledge of the teacher and may even learn to identify classes that are entirely missing from the training data.

When performing KD through this method, it is important to relax the softmax distributions from the teacher and the student so that their pseudo probabilities are more evenly distributed. By doing this, more information will be transferred by the false soft labels through the loss-function. This is achieved by dividing the raw logits with a parameter T (short for temperature) before applying softmax on them. An example of the effect of softening probabilities is given in Figure 4.

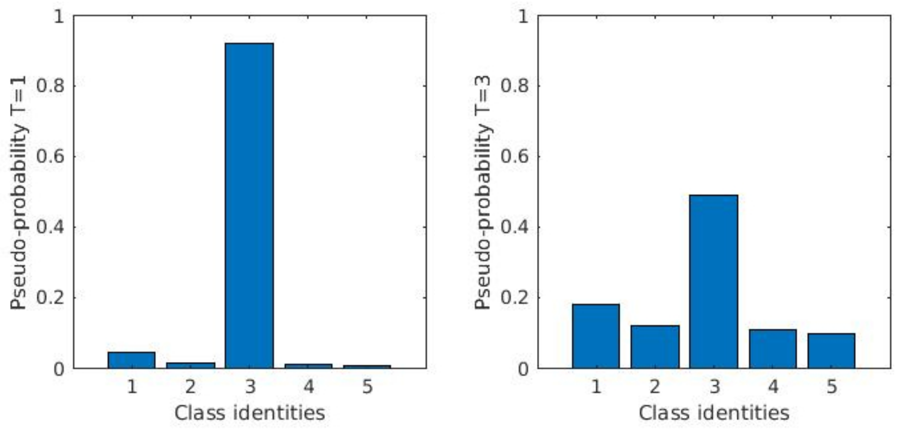


Figure 4: Example of softmax output with raw logits and logits that have been softened with temperature T=3.

If y_{pred} are the predictions by the student and y_s are the soft labels from the teacher, then their respective softened versions can be denoted as y'_{pred} and y'_s. The hard labels, y_h, do not need to be relaxed. If we represent the cross-entropy for the hard labels as E_h and let E_s be the cross-entropy for the soft labels then the Hinton-loss function can be summarized as

$$L_{KD} = \alpha E_h(y_{pred}, y_h) + \beta E_s(y'_{pred}, y'_s) \tag{3}$$

where α is a hyper parameter to balance the two losses during training and β is derived from α as $\beta = 1 - \alpha$ [25].

2.2.2 Knowledge Distillation using Features

A model which performs identification produces a distribution of probabilities for a set of classes. In contrast, a model which performs verification only produces a set of features which are not linked to a class probability. The elements of the distribution are by nature confined to a numerical interval of $[0, 1]$ and they always sum to 1, while the raw features of a network can take any number. This requires a slightly different approach to achieve knowledge distillation on features. Yan et al. [10] presented a loss-function which normalizes the features of the teacher and student network and then calculates the mean-squared error of the difference between them. The loss is expressed as

$$L_{feature} = \frac{1}{N}\sum_{i=1}^{N} || \frac{F_t^i}{||F_t^i||} - \frac{F_s^i}{||F_s^i||} ||_2^2 \tag{4}$$

where F_t and F_s are the teacher and student output features respectively and N is the batch-size. This loss was combined with loss-function (2), that compares the guess by the student to the ground truth label

$$L_{tot} = L_{arc} + \alpha L_{feature} \tag{5}$$

where α is a hyper-parameter to weigh the losses. Yan et al. empirically set $\alpha = 7$ [10].

2.2.3 Staged Knowledge Distillation

When the reduction of parameters or the change of architecture from a teacher model to a student model is very large, the student has to represent the knowledge of the teacher in a much more complex and dense way [10]. This may cause the student to be unable to learn enough during the training. One way to relieve such problems is to carry out the training in two stages using a model of intermediary complexity, as shown in Figure 5. The model in the intermediary stage is known as the adaptor model and will constitute a middle ground between the preceding teacher and the succeeding student in terms of the number of parameters and the architectural changes. The adaptor model will first take on the role of the student and learn from its teacher. When the adaptor model is taught, it can take on the role of the teacher and train the final student. By distilling in two stages, the knowledge is easier for the two students to absorb and this has been shown to further improve the knowledge of the final student [10].

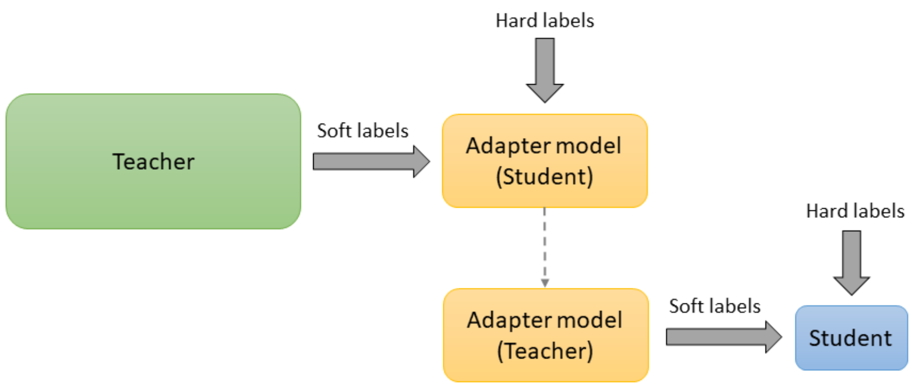


Figure 5: Principle of staged knowledge distillation where the adapter is first taught by the teacher and then takes on the role of teacher for the student model.

2.2.4 Knowledge Distillation using Hints

Most KD approaches try to distill a teacher network into a student network which is thinner (less filters), shallower (fewer layers) or both. Depth (more layers) is however essential for the network's ability to learn [25]. For some families of functions, the network's ability to approximate them grows exponentially with depth. It is, however, more difficult to train a deep network since the same non-convex and non-linear properties that make the network more powerful also make the optimization process more complex. A technique to work around this and to enable deeper and thinner networks that may achieve the same or better performance with fewer parameters has been described as intermediate-level hints [25]. These hints reveal the features of the teacher's hidden layers such that the student can imitate not only the final classification probability but also the method to reach it. Imagine teaching oil painting to a beginner. It is valuable to explain what makes a finished painting good. But for a beginner, it may be just as insightful to reveal the technique to achieve the first few primitive layers of a complex painting.

To achieve this effect in practice, an additional loss function is used [25]. This loss is very similar to the basic KD loss explained in Section 2.2.2, but compares the output from a hidden layer of the teacher and student models, rather than their output layers. Since the goal is to train a student which is thinner (preferably also deeper), the teacher's and student's hidden layer might not have the same size. To handle this, a regressor with the same number of outputs as the teacher's hidden layer is added to the student's hidden layer. This adds parameters but they are hopefully fewer than those removed in the preceding layers. The hint-loss function is written as

$$L_{Hint} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\|F_t - r(F_s)\|^2 \tag{6}$$

where F_t is the feature output from the teacher's hidden layer and $r(F_s)$ is the feature output from the student's hidden layer, transformed through the regressor r.

During training, a stage-wise learning approach is used [25], similar to Section 2.2.3. In the first training stage, only the section up until the hint's regressor is trained such that the student can learn the correct feature mapping to that point. A flowchart of the first stage of the hint training procedure is given in Figure 6. In the second stage, the layers after the regressor are included in training and trained with the normal KD-approach described in Section 2.2. If implemented successfully, this may allow a thin and deep student to express the same or better abilities than that of the thick and deep teacher, but with fewer parameters.

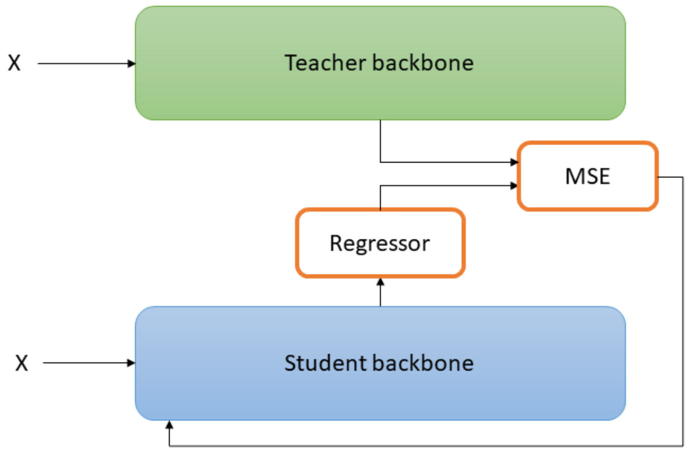


Figure 6: Flowchart describing the first stage of the hint training procedure.

2.3 MobileNetV2

MobileNetV2 is a network architecture developed especially for mobile devices, designed to achieve low computational cost while maintaining high accuracy [26]. The main building block is a bottleneck depth-separable convolution with residuals; the basic structure can be seen in Table 1. The block consists of a 1×1 convolution followed by a 3×3 depthwise separable convolution followed by another 1×1 convolution. The nonlinear activation function Rectified Linear Unit (ReLU) is used in the first two layers.

Table 1: The layers of the bottleneck residual block.

Bottleneck residual block
1x1 Convolution, ReLU6
3x3 Depthwise, ReLU6
Linear 1x1 Convolution

Depthwise separable convolutions consist of two layers, firstly a depthwise convolution and secondly a 1×1 convolution, which are factorized versions of the a normal convolutional operator [26]. Depthwise convolution applies a single convolutional filter per input channel. The second layer, the 1×1 convolution, also called pointwise convolution, computes linear combinations of the input channels and outputs new features. The result of depthwise separable convolutions compared to the normal convolutional layer is a reduction in computational cost by a factor close to k^2, where k is the kernel size, with only a small reduction of accuracy.

Non-linear layers in bottlenecks have been empirically shown to lower the performance of networks [26], therefore, MobileNetV2 uses linear bottleneck layers in its final layer. A residual block creates a skip connection between the beginning and end of a convolutional block, a shortcut surpassing the middle layers, and allows the network to access earlier activations from the convolutional block that were not modified. MobileNetV2 connects the outer narrow layers with a skip connection, surpassing the wide expansion layers in the middle. This approach is called inverted residual since a normal residual block has a skip connection between its wide layers and surpasses the narrow layers.

The modular architecture allows it to be easily scaled to different depths (number of filters in each convolutional layer) using a width-multiplier η. It increases the number of parameters quadratically while maintaining other proportions of the net such as the number of layers and their feature map size. This makes MobileNetV2 suitable for knowledge distillation since models of different sizes can be evaluated with minimal uncertainties due to architectural changes.

3

Methods

This chapter describes the practical approaches to implement the theory described in Chapter 2 as well as the configuration and methodology of the experiments carried out in the project. The implementation of KD for identification and for verification is described in separate sub-chapters to prevent any confusion between the two approaches.

3.1 Data, Data Preprocessing and Augmentation

The data consist of NIR images of faces at a resolution of 128×128. Two sets of data were used for the different experiments, one large unbalanced data set of 4 425 670 images and one small balanced set of 29 997 images. The large data set is used for training and validation of face identification models and training of face verification models. The small data set, which contains new unseen identities, is used for validation of the verification models.

The images in the data set were normalized to a zero mean and unit standard deviation to improve numerical stability and speed up the training process. The NIR images have one channel, similar to gray scale images, but were modified to have three channels by duplicating the single channel input. This modification was needed to fulfill the input requirement of MobileNetV2. Augmentation was performed on the data used for training to improve results; the images were rotated between $\pm$ 5 degrees and translated and scaled between $\pm$ 5 %.

The large data set is highly imbalanced with some identities occurring more regularly then others which causes a bias in the data set. The data is assigned weights based on their occurrence rate such that identities with low occurrence rate receive high importance and vice versa. The small data set is already balanced with each identity appearing with the same frequency.

3.2 Knowledge Distillation for Identification

Knowledge distillation for identification uses the MobileNetV2 architecture configured to have a classifier with size corresponding to the number of classes in the data set. The teacher, adapter and reference models were assigned width multipliers $\eta = 1$, $\eta = 0.68$ and $\eta = 0.01$ resulting in parameter counts 3 529 046, 2 451 270 and 1 335 526 respectively. Since the final classifier is huge, the networks used for identification are far larger than the ones used for verification. Note that the backbone of the small student model consists of only 118 576 parameters when the classifier is excluded. From the database, 110 392 images were split into a validation set, leaving 4 305 278 images for a training set. During validation, the networks were asked to guess the identity of each of the 110 392 pictures where the fraction of correct guesses constitutes the accuracy.

3.2.1 Knowledge Distillation using Hinton Algorithm

The Hinton loss function was implemented between two MobileNets in a teacher-student configuration with 3.53 million and 1.34 million parameters respectively. The architecture of this configuration is given in Figure 7. The teacher model was pre-trained on the data until it converged and was then excluded from the optimization process during the knowledge distillation.

To train with hard labels, the student's logits from the classifier were passed into a loss function constituting of a softmax function followed by cross-entropy operation with the labels. To train for soft labels, both the teacher's and student's logits were softened by the softmax function with temperature t and then passed into the cross-entropy operation.

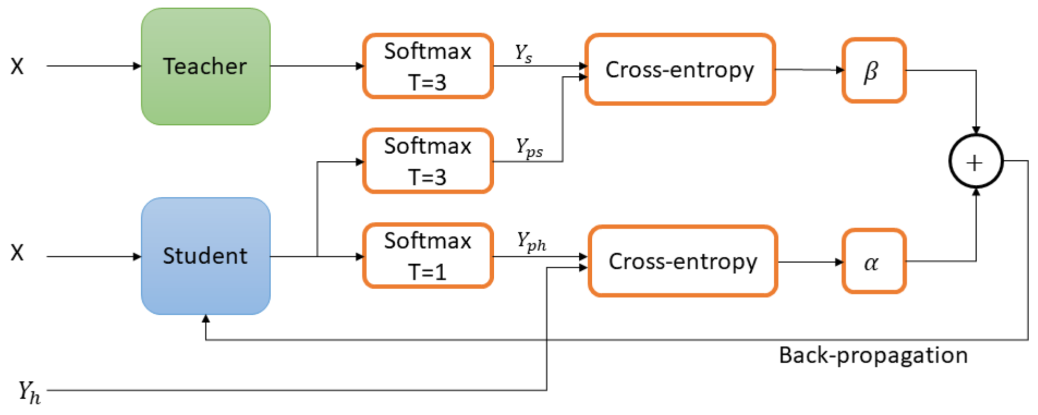


Figure 7: KD algorithm implemented with Hinton loss function. Y_s and Y_{ps} are the soft predictions from the teacher and student respectively. Y_{ph} is the hard prediction from the student.

A parameter study was completed to find the most advantageous parameters α and temperature. Firstly the parameter α was investigated while keeping the temperature value at 3. The accuracy after 15 epochs can be seen in Table 2.

Table 2: Parameter study for α for the identification model.

α	Accuracy (%)
0.1	98.60
0.2	98.71
0.3	98.50

The first study concluded that an α of 0.2 was the most advantageous for the current set up of teacher and student networks. The second parameter study investigated different values of temperature for 15 epochs with α fixed at 0.2.

Table 3: Parameter study for temperature for the identification model

Temperature	Accuracy (%)
2	98.42
3	98.71
4	98.64
5	98.60

As shown in Table 3, the temperatures under and above 3 produce lower accuracy. Therefore, temperature equal to 3 combined with α equal to 0.2 was considered the most advantageous parameter combination for training of students.

3.2.2 Staged Knowledge Distillation

Large discrepancy between the student and teacher model architectures can cause inefficient learning in KD, as described in Section 2.2.3. Staged knowledge distillation is implemented with a 3.53 million parameter teacher, 2.45 million parameter adapter model and 1.34 million parameter student to investigate the effect of dividing the training in two steps. The adapter model is trained using the same teacher as in previous experiments. When the adapter model is fully trained it takes on the role of the teacher and trains the student.

3.2.3 Knowledge Distillation using Hints

A hint based experiment was conducted with a teacher and student model with 3.53 million and 1.34 million parameters respectively. The hint layer in the teacher and the guided layer in the student was chosen to be layer 9, in the middle of the models. At this layer the channel depths were 64 in the teacher and 8 in the student, with the feature map size 8×8 for both models. An additional CNN layer was used as the regressor to bridge the gap in channel depth between the student and the teacher. Mean-squared error was chosen as the loss function for the features, as described in Section 2.2.4. The parameters before the hidden layer are first trained with only feature loss. Following this, the feature loss is disconnected, and the entire student is trained using the Hinton loss and ordinary hard labels loss.

3.3 Knowledge Distillation for Verification

The model used for knowledge distillation in the verification configuration uses the MobileNetV2 architecture as its backbone and two different tops, one for training and one for predictions, as described in Figure 8. The model backbone ends with a fully connected layer of size 64 and uses width multipliers $\eta = 0.6$, $\eta = 0.37$ and $\eta = 0.01$ for the teacher, adapter and student models. This resulted in models of 1 021 920, 493 456 and 118 576 parameters respectively. During training, the output features of the backbone are propagated through a top consisting of the ArcFace layer and loss which places each class in a 64-dimensional spherical space. The design of the training procedure aims to achieve high intra-class compactness and inter-class separability, placing images from the same class close together in the 64-dimensional space and images of different classes far away. During validation, the features are propagated through a top consisting of a normalization layer.

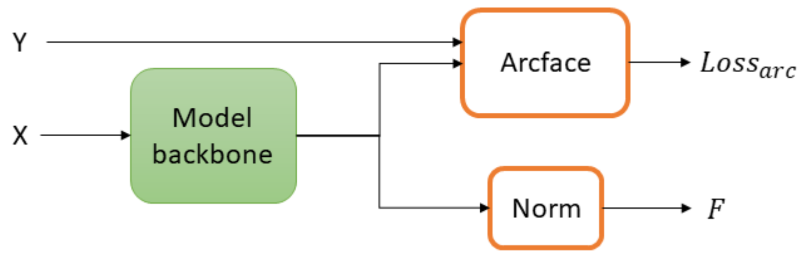


Figure 8: This flowchart describes the two different model tops used for the verification models. The ArcFace layer is used during training and the normalization layer during validation.

The prediction task for verification is to ascertain whether or not two images belong to the same class, i.e. same identity, or not. The output features of the images are normalized, and the cosine distance between the features is calculated. A threshold of the distance is used to determine whether or not a pair of images are a match. The accuracy of the verification model is measured in equal error rate (EER), which is when false acceptance rate (FAR) and false rejection rate (FRR) are equal. The data set of 4.4 million images was used for training and the set of 30K images was used for validation.

3.3.1 Knowledge Distillation using Features

Knowledge distillation was implemented using a total loss (5), based on the features of the teacher and student, $L_{feature}$ combined with the arc face loss, L_{arc}. The features loss uses a basic tensor normalization and mean-squared error operation of the difference between the normalized student and teacher features. The losses are implemented in the teacher-student configuration during KD-training, as shown in Figure 9, and the two losses are weighted by a parameter alpha.

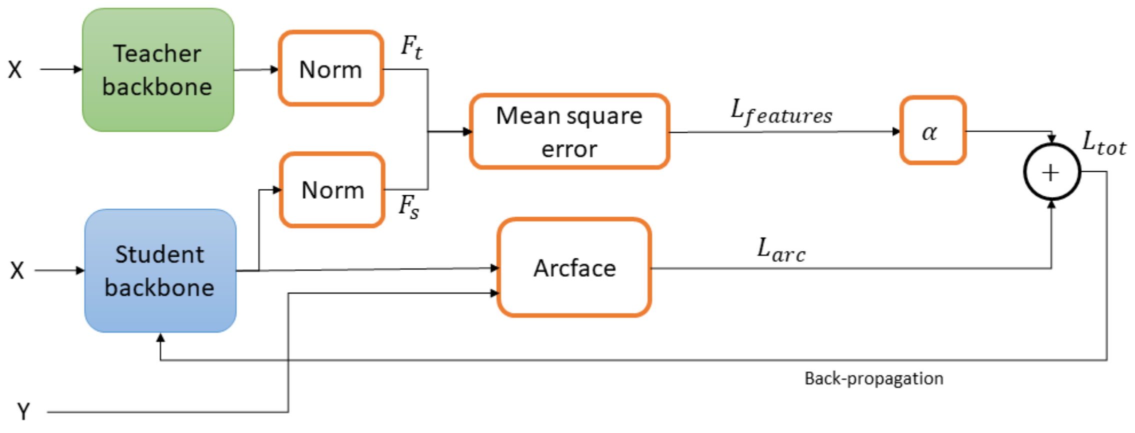


Figure 9: KD algorithm implemented with feature loss, $L_{feature}$, and ArcFace loss, L_{arc}. F_t and F_s are the normalized features of the teacher and student models.

A teacher and a student model of 1.0 million and 119 K parameters respectively, were used. The teacher model was pre-trained until learning stoped improving. A parameter study was done to determine a suitable value of alpha. Four different values of α were chosen and trained for three epochs, see results in Table 4. The α values chosen for the study were based around the implementation by [10], which pinpointed at an α of seven to be successful.

Table 4: Parameter study for α for the verification model.

α	EER (%)
5	5.867
6	4.994
7	5.013
8	5.251

The results of the study indicate an α of six to be the optimal value with higher EER for alphas of both lower and higher values. Therefore, the α of six was used for training of the student models.

3.3.2 Staged Knowledge Distillation

Staged KD was implemented between a teacher model of size 1.0 million, an adapter model of 493 K parameters and a student model of 119 K parameters. The models are trained in the 2 step procedure described in Section 2.2.3, teacher distilling knowledge to the adapter and the adapter distilling knowledge to the student.

3.3.3 Knowledge Distillation using Hints

The first half of a student model of 119 K parameters was pre-trained with a teacher model of 1 million parameters. This was done by using the loss for features with a regressor at layer 9 in both models, as described in Section 2.2.4. At this layer the channel depth of the teacher and student was 40 and 8 respectively. Both models had the same feature size of 8×8. The regressor, a CNN-layer, is used to bridge the gap in channel depth between the teacher and student. After this the entire student was trained using the loss for normalized features and a labels loss, according to procedure described in (4).

4

Results and Discussion

The results of the experiments described in the Methods chapter are presented and discussed. Firstly, the results of KD for identification are presented and discussed followed by KD for verification. Lastly, a discussion comparing the different KD methods and related results.

4.1 Knowledge Distillation for Identification

The results of the different KD methods used for identification is discussed starting off with the Hinton implementation on the NIR-data, followed by the results of Staged KD and KD using hints.

4.1.1 Hinton Algorithm

The basic Hinton implementation was applied to the NIR data with a 3.53 million parameter teacher and a 1.34 million parameter student, and the results are given in Table 5. The student was trained with α set to 0.2 and the temperature set to 3. This resulted in a student which performed only slightly better than its reference. Some earlier works [7] [27] spoke of difficulties to successfully train a student when the difference in size between teacher and student is large, which might be why the improvement was dissatisfying. We speculate that the teacher is too complex for the student to replicate.

An experiment with alpha set to 0 was carried out to investigate the effect of training without the ground truth labels. With alpha set to 0, the hard labels, y_h, are left out of the loss function and the training only tries to optimize the soft loss, E_s. It was shown that with this data, removing the loss for hard labels made the knowledge distillation perform poorly. The accuracy was lower than both the reference model trained without KD as well as the model trained with true labels, indicating that the true labels were of high importance.

Table 5: Results for identification using the Hinton algorithm.

	Parameters	Accuracy (%)
Teacher	3.53 M	99.96
Student Reference	1.34 M	99.37
Student	1.34 M	99.43
Student (No labels)	1.34 M	99.36

4.1.2 Staged Knowledge Distillation

To overcome the problems with large differences in size between the teacher and student, a staged method of training was tried. The student was trained with α of 0.2 and temperature of 3. KD using a staged training procedure was not successful in the case of identification. As shown in Table 6, the staged student model reached an accuracy equivalent of the reference model but did not improve any further. The accuracy of the adapter model did not outperform the reference adapter model which might be one reason for an unsuccessful staged training.

Table 6: Results for staged KD for identification.

	Parameters	Accuracy (%)
Teacher	3.53 M	99.96
Adapter Reference	2.45 M	99.96
Adapter	2.45 M	99.95
Student Reference	1.34 M	99.37
Student (Staged)	1.34 M	99.37

4.1.3 Knowledge Distillation using Hints

Another experiment was conducted to determine the effect of a hint based training, where parts of the student is pre-trained. The parameters up until the hidden layer were first trained with feature loss for 10 epochs. Then the student was trained until reaching maximum capacity using Hinton loss and ordinary labels loss with α and temperature set to 0.2 and 3 respectively. This resulted in a student which performed better than the student reference and better than the student trained with only Hinton KD, results shown in Table 7. This proves that there is an advantage in pre-training parts of the model. The hint based training seem to relieve the problem of large size difference between the teacher and student.

Table 7: Results for hints training for identification.

	Parameters	Accuracy (%)
Teacher	3.53 M	99.96
Student Reference	1.34 M	99.37
Student (KD w. Hints)	1.34 M	99.52

4.2 Knowledge Distillation for Verification

In the following chapter the results of KD for verification is presented and discussed. Firstly KD using features is presented and thereafter the results for staged KD and KD using hints.

4.2.1 Knowledge Distillation using Features

KD was implemented using a feature based loss and resulted in improved accuracy for the student compared to the reference when distilling knowledge from the 1 million parameter teacher to the 119 K parameter student, as shown in Table 8. The weighing parameter α was set to 6. There was an improvement in EER between the reference model and student model, taking the student half the way to the performance of the teacher, the improvement is visualised using a Receiver Operator Curve (ROC) in Figure 10, showing the relation between FRR, FAR and EER.

Table 8: Results for KD using features for verification.

	Parameters	EER (%)
Teacher	1 M	3.12
Reference	119 K	4.38
Student	119 K	3.78
Student (No labels)	119 K	4.04

Next, an experiment to investigate the effect of training without the ground truth labels, similar to the one in the identification chapter, was carried out by disconnecting the L_{arc} loss and final fully connected layer of the student. This means that the loss function only considers the feature loss $L_{feature}$ and not the ground truth labels. The student trained without ground truth labels achieved a higher performance than the reference model but did not beat the performance compared to the one using labels.

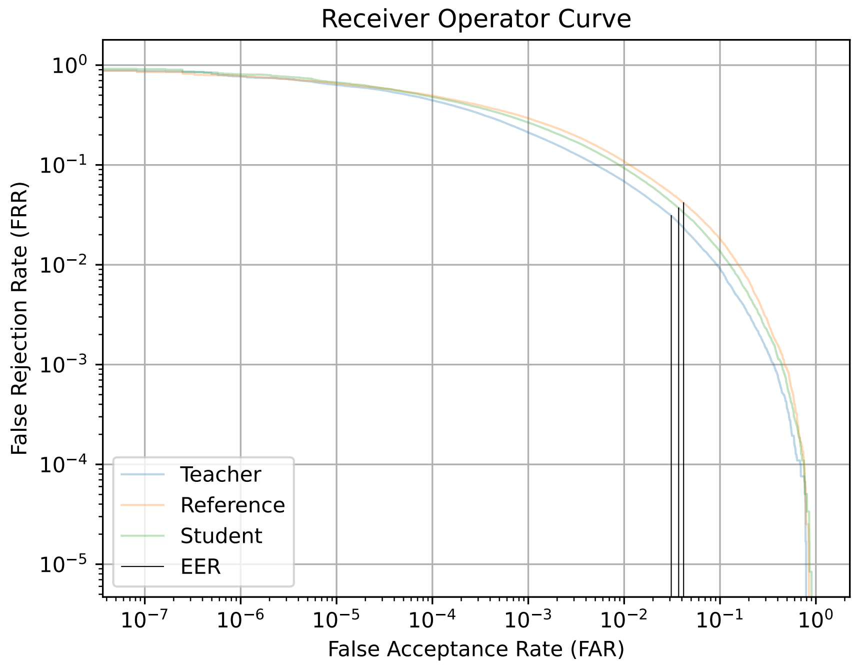


Figure 10: The ROC diagram of of the teacher, student and reference models for KD using features. The EER is marked out for each model.

4.2.2 Staged Knowledge Distillation

Staged KD was used in an attempt to improve the training between a teacher and student with a large discrepancy in size. During training α was set to 6. The student trained using the adapter model achieved an improved accuracy compared to its reference, as shown in Table 9. The relation between FRR, FAR and EER for the staged student is shown in Figure 11. There was also an improvement in accuracy when training in a two step procedure compared to KD in one step. This shows that a staged training can further improve the performance of KD.

Table 9: Results for staged KD for verification.

	Parameters	EER (%)
Teacher	1 M	3.12
Adapter Reference	493 K	3.15
Adapter	493 K	3.08
Student Reference	119 K	4.38
Student (Staged)	119 K	3.69

An interesting result is that the adapter model performed better than its teacher. This might be due to the fact the teacher was only slightly better than the adapter reference model, therefore, the model was able to reach a high accuracy already without KD. With the addition of KD, the training was made more efficient and the knowledge compressed, allowing for the adapter model to absorb more knowledge.

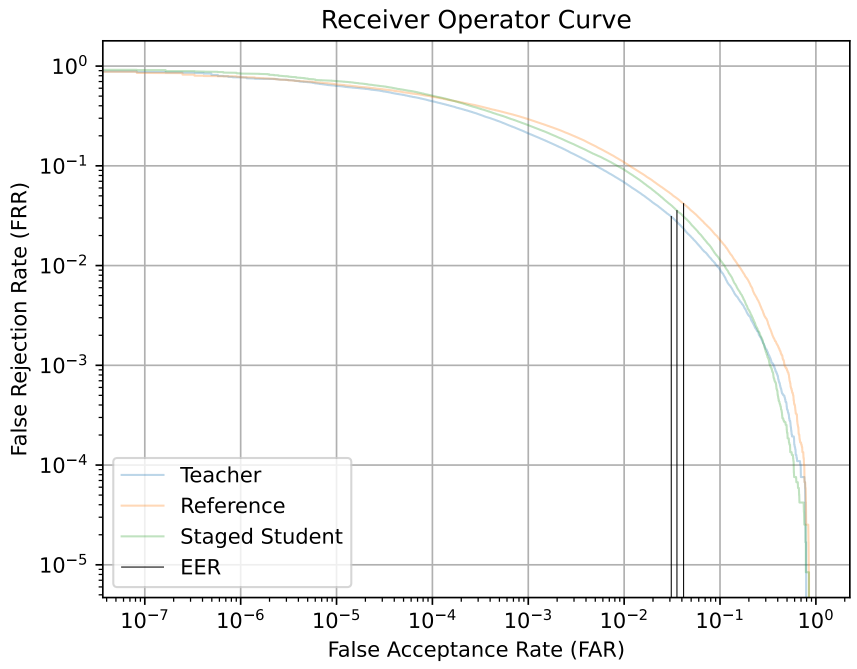


Figure 11: The ROC diagram of of the teacher, staged student and reference models for staged KD. EER is marked out for each model.

The small difference in EER between the adapter reference and teacher indicates that the increase in parameters did not result in a model of much higher accuracy. This may be due to the fact that a large model might need more training data to be able to make use of all the parameters.

4.2.3 Knowledge Distillation using Hints

The hint training was implemented on the verification problem with a 10 epoch hints training of the student model up to its middle layer, then continued with the standard training procedure of the whole model using the feature loss. This resulted in a student which did achieve a lower EER than its reference without KD, as shown in Table 10 and in Figure 12. The student trained with hints was only slightly better than the method without hints. Further tuning of parameters and number of epochs for the hint training might improve results further.

Table 10: Results for hints training for verification.

	Parameters	EER (%)
Teacher	1 M	3.12
Reference	119 K	4.38
Student (Hints)	119 K	3.75

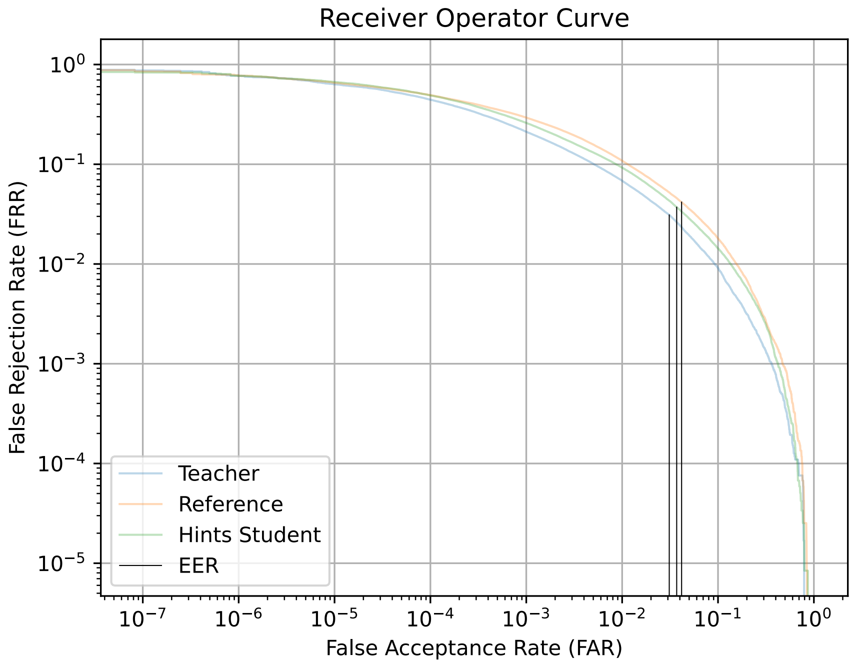


Figure 12: The ROC diagram of of the teacher, Hints student and reference models for KD with hints. EER is marked out for each model.

4.3 Comparison of the Methods

The results show that KD generally works for both the case of identification and verification, as shown in Table 11. The basic KD implementations achieved students with a higher accuracy than the reference for both identification and verification.

The experiments of hints and staged show that these training procedures are useful to bridge the gap between a large teacher and a small student. In the case of identification staged training was not successful but using hints training the student managed to achieve higher accuracy than with the standard KD procedure. An improved learning is also noted in the case of verification where staged and hints managed to outperform the student trained with only KD.

Staged training procedure was only successful for verification and not for identification, this might be due to a need of optimization of model sizes in the stages. Another possible reason could be that the case of identification was unfit for staged KD due to the small differences in accuracy between the small and the large models. There is only a small difference in accuracy between the models, all reaching an accuracy of 99 %, even without KD. The addition of KD brings on small improvements but nothing of great significance. We speculate that the identification case might have been unsuitable for KD since there was not a lot of room for improvement in model accuracy.

The experiment of removing the label loss proved the importance of the labels during training. Labels seemed especially important for the identification case where the model trained without labels did not improve accuracy at all compared to the reference model.

Table 11: Results for the different methods. Accuracy for identification and EER for verification.

	Accuracy (%)	EER (%)
Student Reference	99.37	4.38
Student	99.43	3.78
Student (No labels)	99.36	4.04
Student (Staged)	99.37	3.69
Student (Hints)	99.52	3.75

4.4 Uncertainties and Sources of Error

When training a neural network, it is often difficult to know when the network has finished learning. During this project, validation was performed after each epoch and when the graph of validation results flattened, the decision to stop the training was taken. It is, of course, possible that improvements to the model could still have been made with an even smaller learning rate in a later epoch.

The NIR data set used to train the models is known to contain errors such as wrong labels and images of faces that should not be in the database. It is unknown to what extent this affects the project. There can be obvious negative effects such as the networks learning wrong information, but it was also theorized that the errors could add a beneficial regularization effect to make the networks more robust. Another possibility is that the errors help improve the KD algorithms performance relative to traditional training. This could happen because the teacher model may produce the correct label for some of the falsely labeled images such that the student has better labeled data to learn from compared to the reference model. Another possible source of error is that the images are taken as frames from video, therefore some frames may be very similar to one another. This might cause an over estimation of the network accuracy in the case of identification which does not use new identities during validation.

The teacher and reference models were tuned and trained to the best possible performance to make for a fair evaluation of KD. The basic KD algorithms Hinton KD from Section 2.2.1 and features KD from Section 2.2.2 were also tuned since they include relatively few hyperparameters. The values for α and T, that were found in the parameter studies for the Hinton and features algorithms, were also used in the staged and hints algorithms but should ideally have been re-tuned. Other parameters used in the staged and hints algorithms were not tuned at all. For staged it was for example the number of stages and the size of each stage and for hints it was the number of hints, their locations in the model and the number of epochs during the first training phase. Since this project sought to evaluate if KD algorithms can improve the performance of light weight models trained on low resolution NIR data, not to find the best performance of each algorithm, this was not regarded as a problem when the algorithms had already shown significant increase in performance over traditional training. It is however important to remember that it is not suitable to draw conclusions about which algorithm performs the best from these results.

5

Conclusion

In devices such as cars and mobile phones, there is a need for small CNN models with low computational complexity which still achieve high accuracy. Knowledge distillation, a method for distilling knowledge from a large teacher model to a small student model, is one way to improve performance of light weight models. This thesis investigates how different knowledge distillation methods can be used to improve the performance of a light weight CNN model for face recognition tasks using low quality near infrared images. The result shows that the usage of knowledge distillation during training can indeed improve learning and increase the accuracy of small CNN models trained on low quality near infrared data. Therefore, it is a useful tool to improve the performance of light weight networks in low computational cost systems. Staged training or using hints during training has been proven to improve the training procedure and accuracy of the student further, especially if there is a large discrepancy in size between the teacher and student.

For future work it would be interesting to tune the hyperparameters of the staged and hints algorithms. This would give us an idea of their absolute performance and perhaps also during which circumstances one may be superior. Combining algorithms is another interesting topic and requires more attention as it might give an even stronger distillation effect. It is indeed interesting to have a training process with several stages, where each stage is trained using one or several hints. Ideally it would also be interesting to try to improve the performance by expanding the depth of the architecture between stages in the way FitNets managed to using hints.

Bibliography

[1] J. Tol, “Ethical implications of face recognition tasks in law enforcement,” diploma thesis, University of Amsterdam, July 2019.

[2] J. Deng, J. Guo, D. Zhang, Y. Deng, X. Lu, and S. Shi, “Lightweight face recognition challenge,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[3] M. Yan, M. Zhao, Z. Xu, Q. Zhang, G. Wang, and Z. Su, “Vargfacenet: An efficient variable group convolutional neural network for lightweight face recognition,” in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[4] X. Zeng and T. R. Martinez, “Using a neural networks to approximate an ensemble of classifiers,” in *Neural Processing Letters*, 2000, p. 2000.

[5] C. Bucila, R. Caruana, and A. Niculescu-Mizil, “Model compression,” vol. 2006, 08 2006, pp. 535–541.

[6] L. J. Ba and R. Caruana, “Do deep nets really need to be deep?” *CoRR*, vol. abs/1312.6184, 2013. [Online]. Available: http://arxiv.org/abs/1312.6184

[7] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015, cite arxiv:1503.02531Comment: NIPS 2014 Deep Learning Workshop. [Online]. Available: http://arxiv.org/abs/1503.02531

[8] J. Gou, B. Yu, S. Maybank, and D. Tao, “Knowledge distillation: A survey,” 06 2020.

[9] S. Ge, S. Zhao, C. Li, and J. Li, “Low-resolution face recognition in the wild via selective knowledge distillation,” *CoRR*, vol. abs/1811.09998, 2018. [Online]. Available: http://arxiv.org/abs/1811.09998

[10] M. Yan, M. Zhao, Z. Xu, Q. Zhang, G. Wang, and Z. Su, “Vargfacenet: An efficient variable group convolutional neural network for lightweight face recognition,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[11] X. Peng, Y.-x. Li, X. Wei, J. Luo, and Y. Murphey, "Rgb-nir image categorization with prior knowledge transfer," *EURASIP Journal on Image and Video Processing*, vol. 2018, p. 149, 12 2018.

[12] X. Wu, R. He, Y. Hu, and Z. Sun, "Learning an evolutionary embedding via massive knowledge distillation," *International Journal of Computer Vision*, vol. 128, 09 2020.

[13] N. Martinez-Martin, "What are important ethical implications of using facial recognition technology in health care?" *AMA Journal of Ethics*, vol. 21, no. 2, pp. ss 180–187, February 2019.

[14] T. U. Nations, "The 17 sustainable development goals." [Online]. Available: https://sdgs.un.org/goals

[15] Biometrics and F. E. Group, "Ethical principles to guide police facial recognition trials," February 2019. [Online]. Available: https://www.gov.uk/government/news/ethical-principles-to-guide-police-facial-recognition-trials

[16] S.-C. Wang, *Artificial Neural Network*. Boston, MA: Springer US, 2003, pp. 81–100. [Online]. Available: https://doi.org/10.1007/978-1-4615-0377-4_5

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[18] M. A. Nielsen, "Neural networks and deep learning," 2018. [Online]. Available: http://neuralnetworksanddeeplearning.com/

[19] G. Antipov, S.-A. Berrani, and J.-L. Dugelay, "Minimalistic cnn-based ensemble model for gender prediction from face images," *Pattern Recognition Letters*, vol. 70, pp. 59 – 65, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865515003979

[20] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," *CoRR*, vol. abs/1704.08063, 2017. [Online]. Available: http://arxiv.org/abs/1704.08063

[21] A. Ross and A. Jain, "Multimodal biometrics: An overview," pp. 1221–1224, 10 2004.

[22] J. Deng, J. Guo, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," *CoRR*, vol. abs/1801.07698, 2018. [Online]. Available: http://arxiv.org/abs/1801.07698

[23] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," *CoRR*, vol. abs/1710.09282, 2017. [Online]. Available: http://arxiv.org/abs/1710.09282

[24] Y. Lecun, J. Denker, and S. Solla, "Optimal brain damage," vol. 2, 01 1989, pp. 598–605.

[25] A. Romero, N. Ballas, S. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *CoRR*, vol. abs/1412.6550, 2015.

[26] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2019.

[27] H. Ni, J. Shen, and C. Yuan, "Enhanced knowledge distillation for face recognition," in *2019 IEEE Intl Conf on Parallel Distributed Processing with Applications, Big Data Cloud Computing, Sustainable Computing Communications, Social Computing Networking (ISPA/BDCloud/SocialCom/SustainCom)*, 2019, pp. 1441–1444.