Proceedings of the 7th International Workshop on Sign Language Translation and Avatar Technology (SLTAT 7), pages 29–38
Language Resources and Evaluation Conference (LREC 2022), Marseille, 20-25 June 2022

Fine-tuning of Convolutional Neural Networks for the Recognition of Facial Expressions in Sign Language Video Samples

Neha Deshpande[1], **Fabrizio Nunnari**[2], **Eleftherios Avramidis**[3]
[1]Technical University of Berlin,
[2]German Research Center for Artificial Intelligence (DFKI),
Saarland Informatics Campus D3.2, Saarbrücken, Germany
[3]German Research Center for Artificial Intelligence (DFKI), Alt Moabit 91c, Berlin, Germany
nehadeshpande97@gmail.com, fabrizio.nunnari@dfki.de, eleftherios.avramidis@dfki.de

Abstract

In this paper, we investigate the capability of convolutional neural networks to recognize in sign language video frames the six basic Ekman facial expressions for 'fear', 'disgust', 'surprise', 'sadness', 'happiness' and 'anger' along with the 'neutral' class. Given the limited amount of annotated facial expression data for the sign language domain, we started from a model pre-trained on general-purpose facial expression datasets and we applied various machine learning techniques such as fine-tuning, data augmentation, class balancing, as well as image preprocessing to reach a better accuracy. The models were evaluated using K-fold cross-validation to get more accurate conclusions. Through our experiments we demonstrate that fine-tuning a pre-trained model along with data augmentation by horizontally flipping images and image normalization, helps in providing the best accuracy on the sign language dataset. The best setting achieves satisfactory classification accuracy, comparable to state-of-the-art systems in generic facial expression recognition. Experiments were performed using different combinations of the above-mentioned techniques based on two different architectures, namely MobileNet and EfficientNet, and is deemed that both architectures seem equally suitable for the purpose of fine-tuning, whereas class balancing is discouraged.

Keywords: facial expression recognition, sign language

1. Introduction

While people are speaking, their facial expressions convey emotional information. Sign languages are visual languages that relies on movements of hands, body, as well as facial muscles. Thus, facial expressions are already involved in conveying the meaning of a message. To what extent, and how, facial expressions of signers are also involved in the communication of emotions is still an open and under-investigated topic.

This work consists of a focused experimentation which is a preliminary step in the broader research on SL recognition, where we try to understand if a computer can recognize facial expressions from a signer as good as it can already do for the facial expressions of speaking subjects. Since this is one of the first experiments on this topic, and given the lack of more descriptive datasets of appropriate size, we hypothesize on the applicability of deep learning and proceed with specific assumptions: we are based on a shallow labelling of only 6 emotions, we don't consider linguistic content/markers and we focus on the face, ignoring spatial and manual elements.

Facial expressions are culture-specific, due to which most positive emotions are communicated with culture-specific signals, while the negative emotions can be recognized across cultures (Sauter et al., 2010). In this work, we focus on German sign language.

Deep convolutional neural networks (CNN), the state-of-the-art in image recognition, require a large amount of data and a limited amount of facial expressions data is available specifically for the German SL, making it difficult to train a Facial Expression Recognition (FER) model from scratch. Therefore, this work uses fine-tuning of pre-trained models that showed a state-of-the-art accuracy on common facial expression datasets. The pre-trained models used during the experiments follow a lightweight architecture which makes it easier to fine-tune and still provides high accuracy.

For this study, it was hypothesized that fine-tuning a pre-trained FER model (trained on a very large image dataset) helps improve the prediction rate on a SL dataset, annotated with the six basic emotions of 'sad', 'surprise', 'fear', 'angry', 'disgust', and 'happy' along with the 'neutral' and 'none' labels. Apart from fine-tuning, the experiments include various machine learning techniques such as data augmentation, image normalization and class balancing to improve the performance of the fine-tuned model.

The rest of the paper is organized as follows. A survey of related literature is given in Section 2. Section 3 includes a description of the methods used. Section 4 contains details about the experiments. Section 5 presents the results of these experiments followed by Section 6, which concludes the paper.

2. Related Work

As discussed in the previous section, to tackle the complexity of FER, several machine learning (ML) techniques have been used including both conventional as well as deep-learning-based approaches. A review of FER in the past years, including a comparison of several techniques based on certain evaluation metrics, is provided in Ko (2018).

Deep-learning-based approaches such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) can perform end-to-end feature extraction, classification as well as recognition tasks with high accuracy (Kim et al., 2019; Chu et al., 2017). However, they need large datasets, computing power, amounts of memory and are time-consuming for both the training and testing phases (Ko, 2018). In the reminder of this section, we introduce related work in the detection of facial expressions using CNNs and how to improve their performance when data is scarce.

2.1. Existing Deep-Learning-Based Models

State-of-the-art techniques involving deep-learning-based approaches used for FER are presented below. Savchenko (2021) presented a simple training pipeline where a model can provide state-of-the-art accuracy using lightweight neural networks in FER trained on images and videos of the AffectNet data-set (Mollahosseini et al., 2019). The high performance, reduced speed and model size of this model is the result of pre-training of facial feature extractor for face identification, which was done by a very large VGGFace2 (Gennaro and Vairo, 2019) data-set. The features extracted by this network can be used with more complex classifiers, and therefore can be explored for FER in the case of SL.

Frame Attention Networks (FAN) can be used to automatically discriminate frames in the network by taking a videos with various image frames as its input and produce a fixed-dimension feature representation which can be then used for FER through a CNN (Meng et al., 2019). This framework provided a high performance on the CK+ (Lucey et al., 2010) and AFEW 8.0 (Kossaifi et al., 2017) datasets (both including seven emotion labels).

Along with deep-learning-based models, models pre-trained with Local Binary Patterns (LBP) to extract facial features and Support Vector Machines (SVM) to classify them were also recently used, although their accuracy on some datasets has been lower than that with a CNN (Ravi et al., 2020).

2.2. Improving CNNs

Data augmentation techniques (O'Mahony et al., 2019), which include geometric transformations such as flipping the training images horizontally, as well as cropping them randomly to increase the training data, are used for improving the performance of a CNN (Savchenko, 2021). In most CV tasks involving image classification, flipping the images horizontally before training is sufficient and helps in improving the overall performance of the CNN (Zheng et al., 2020). Apart from data augmentation, using data preprocessing techniques such as resizing, face detection, cropping, adding noise, data normalization, histogram equalization, etc., also helps in boosting the performance of a CNN trained for recognizing emotions from facial images (Pitaloka et al., 2017).

CNN architectures such as EfficientNet (B0 to B7), MobileNet, ResNet, etc., help in reducing the calculations required making them more lightweight and faster (Tan and Le, 2019; Tan and Le, 2021). Using several optimizers instead of just one also improves the performance and generalization of a CNN (Taqi et al., 2018). Along with the commonly used Adam optimizer, using an additional optimizer such as the Sharpness Aware Minimization (SAM) and Stochastic Gradient Descent (SGD) for the last few epochs boosts the overall performance by providing a better coverage (Savchenko, 2021). Other important parameters that could help boost the performance of a CNN are: appropriate learning rates, choice of the activation function, balancing the imbalanced classes, etc. (Kandel and Castelli, 2020).

Last but not least, transfer learning, i.e., using the parameters learned for a problem as starting values (instead of random values) to train on a new dataset, helps in reducing training time (Akhand et al., 2021; O'Mahony et al., 2019).

3. Methods

This section describes the methods used in the experiments. The methods explained are chosen due to the state-of-the-art accuracy they provided in Facial Expression Recognition models presented by Savchenko (2021).

3.1. Image Preprocessing

Image preprocessing plays a vital role in achieving state-of-the-art results in a CNN, as the raw data does not always produce good accuracy. The improvement in accuracy of a CNN is dependent on the image preprocessing technique being used along with its network architecture. This work uses two image preprocessing techniques: face cropping and image normalization.

Face Cropping is a technique used in CV to extract the area of the image which is required for image recognition or classification tasks. In the case of FER, faces are cropped from the image dataset to remove the unnecessary information from the images and only keep the pixels that constitute the facial information. To crop faces from an image, Savchenko (2021) has proposed the use of a Multi-task Cascaded Convolutional Network (MTCNN), a framework used for face detection and alignment. MTCNN performs three tasks: face classification, bounding box regression, and facial landmark localization (Xiang and Zhu, 2017).

The face is cropped from an image in the following steps: detect and extract the face mesh from images, extract the face bounds, and then crop the images (Emami and Suciu, 2012). Detection and recognition of faces is done using a Haar Cascade (Soo, 2014), an object detection method used to locate an object of interest in images. A Haar-like feature considers neighboring rectangular regions, sums up the pixel intensities in each region, and calculates the difference between these sums, which helps to categorize the image into subsections. We use the implementation of OpenCV, which has shown good performance for face detection (Boyko et al., 2018).

Image Normalization. Studies have shown that for image classification as well as recognition tasks image normalization has helped in enhancing the performance of the CNN (Savchenko, 2021; Koo and Cha, 2017; Heidari et al., 2020).
Image normalization is a technique where the mean along each of the features (dimensions of images) from the training sample is calculated and is subtracted from every image. This results in normalizing the brightness of the whole training set concerning each dimension as shown in the equation below (Pal and Sudeep, 2016):

$$X' = X - \mu \tag{1}$$

where X' is the normalized data, X represents the original data, and μ is the mean vector across all features of X.

3.2. CNN Architectures

CNNs are artificial neural networks that play a significant role in Natural Language Processing (NLP), CV tasks such as image detection, recognition, etc. (Albawi et al., 2017). Several CNN architectures have been developed to solve real-world problems. In this work we use the MobileNet and EfficientNet architectures which are explained below.

The MobileNet Architecture - MobileNet-v1 The MobileNet architecture (Savchenko, 2021) uses depthwise separable convolutions followed by pointwise convolutions where each input channel is filtered separately. This results in a drastic reduction in model size and cost compared to standard convolutions. In comparison to other more efficient architectures, the accuracy obtained with MobileNet reduces as the number of parameters is increased in the model.
The MobileNet v1 architecture has 28 layers wherein each layer is followed by batch normalization and a Rectified Linear Unit (ReLU) (Ioffe and Szegedy, 2015). The architecture starts with a regular 3×3 convolution, followed by 13 depthwise separable convolutional blocks and pointwise convolutions (Michele et al., 2019). The depthwise convolution in MobileNet is the channel-wise spatial convolution (Howard et al., 2017). Whereas the pointwise convolution is 1x1 convolution which is used to change the dimension. These depthwise and pointwise convolutions result in a reduction in model size and computation cost by about 8 to 9 times as compared to the usage of standard convolutions (Sinha and El-Sharkawy, 2019).
The MobileNet v1 architecture has been used for a variety of object detection and image recognition applications such as palm print recognition (Michele et al., 2019), handwriting character recognition (Ghosh et al., 2020), FER (Savchenko, 2021), and more.

The EfficientNet Architecture - EfficientNet-B0 EfficientNet (Tan and Le, 2019) is another neural network architecture that consists of 8 model types, from B0 to B7. The accuracy and the number of model parameters increase with the model number. EfficientNet uses an activation function called Swish instead of the Rectifier Linear Unit (ReLU) of the MobileNet architecture. The main building block for EfficientNet is the inverted bottleneck MBConv, which consists of a layer that first expands and then compresses the channel (Tan and Le, 2019; Sandler et al., 2018). This architecture has in-depth separable convolutions that reduce the calculation by almost k^2 factor compared to traditional layers, where k is the kernel size which denotes the width and height of the 2D convolution window (Sandler et al., 2018). EfficientNet has been recently used for several applications such as plant leaf disease classification (Atila et al., 2021) and automated diagnosis of COVID-19 (Marques et al., 2020).

The EfficientNet architecture is more efficient than MobileNet and has provided state-of-the-art accuracy on several transfer learning datasets as it is easily scalable (Tan and Le, 2019). On the one hand, when used for image classification problems, the EfficientNet architecture scaled up the image size leading to large memory consumption compared to MobileNet. On the other hand, the MobileNet architecture is more lightweight and it works efficiently for a small number of parameters.

3.3. Training

Fine-tuning is the process of initializing a pre-trained classification network and then training it further for a different task (Radenović et al., 2018). It is applied when there is the need to fit a low resource dataset starting from models pre-trained on bigger datasets. One of the motivations for using fine-tuning instead of fully training a model from scratch is that the low-level basic features are common for most images and hence an already trained (pre-trained) model can be useful for classification by just fine-tuning the high-level features.
The proposed FER technique (Akhand et al., 2021; Ngo and Yoon, 2020; Savchenko, 2021) is to use a CNN model pre-trained for image classification, and fine-tune it by replacing the upper layers with the dense layer(s) to make it compatible with the target dataset. These new dense layers are first tuned to the target dataset, followed by training the whole CNN with this same dataset.

Optimization in Neural Networks The aim of a CNN is to learn from the given data by minimizing the loss. The loss function is reduced with the help of an optimization algorithm which is a numerical function performed on the model parameters. A gradient descent algorithm is commonly used in neural networks for optimization as it minimizes the objective function by updating the parameters in the reverse direction of the gradient of the objective function. Here we briefly explain the three optimizers used. In addition to the most popular optimizers (such as Adam and Stocastic Gradient Descent), **Sharpness Aware Minimization (SAM)** is an optimization technique that seeks parameters that lie in neighborhoods having uniformly low loss leading to sub-optimal model quality (Foret et al., 2020). SAM is shown to improve the generalizability of the model across several datasets and to provide robustness to noisy labels and helped achieve a better performance when applied on fine-tuned EfficientNet models pre-trained on ImageNet. Using SAM for optimizing the categorical cross-entropy loss for the last two epochs also provided a state-of-the-art accuracy on fine-tuned EfficientNet models pre-trained on ImageNet (Savchenko, 2021).

Data Augmentation Flipping data horizontally before feeding it to the CNN has been shown to be not only safe but also one of the most common and effective data augmentation technique (Shorten and Khoshgoftaar, 2019). Other augmentation methods, such as rotation and noise disturbance are not used here, because as noted by Zheng et al. (2020), they could have a large impact on the image structure if the images are small in size, resulting in poor performance.

Class Weights The datasets available for FER do not always consist of balanced classes as they have a different number of samples in each class. This can result in incorrect evaluation and a need for balancing these classes to achieve uniform results across classes. An algorithm-based technique used to balance the classes is called class weighting where different weights are used for every class depending on the number of training samples present in a class. As explained by Johnson and Khoshgoftaar (2019), class weights for each class can be calculated as follows:

$$cw = max_i|Ci|min_i|Ci|$$

Here, cw is the class weight for a minority class. Consider that the largest class in the dataset has 100 samples and the smallest class has 10 samples. If the class weight for the majority class is set to 1 then that for the minority class will be set to 10.

4. Experimental Setup

The goal of our experiments is to maximize the classification accuracy of the facial expressions. As baseline and a basic model for fine-tuning we used the models by Savchenko (2021), trained on generic facial expression data, as they provided a state-of-the-art accuracy with the MobileNet and EfficientNet architectures. In our experiments, the baseline model is first fine-tuned to a dataset of facial expressions of signers. Then, various techniques such as data augmentation, image pre-processing, as well as class weight balancing were applied one after the other in different combinations.
For every configuration we measure the overall accuracy, the sensitivity per class, as well as the average sensitivity. The overall accuracy is the ratio of the number of correct predictions to all the predictions, whereas the sensitivity per class gives the ratio of the correct predictions of a class over its number of samples.

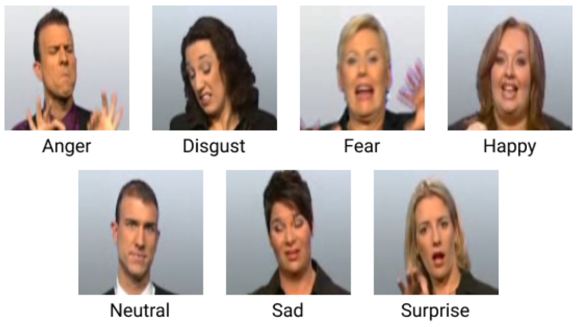

Figure 1: Images from 7 classes in the FePh dataset

4.1. Sign Language Dataset

The fine-tuning was targeted on the Facial Expression Phoenix (FePh) dataset (Alaghband et al., 2021), an annotated sequenced facial expression dataset in the context of the German SL. It comprises over 3,000 facial images extracted from the daily news and weather forecast of the public TV-station PHOENIX. The data was annotated with the six basic Ekman (1999) emotions of 'anger', 'disgust', 'fear', 'sad', 'happy', and 'surprise' along with the 'neutral' class (see figure 1). An aditional 'none of the above' class exists for images where no label could be assigned. Known limitations of this dataset are the size of the dataset, the existence of only 6 shallow labels, the lack of linguistic/content markers and the lack of spatial and manual elements. Since to the best of our knowledge this was the only available dataset suitable for this task, we proceed with using it despite the mentioned concerns in order to confirm our technical hypothesis.

4.2. Data Preparation

The FePh dataset went under three pre-processing steps. The first step consisted of **removing frames** of two types. The first type is the frames labeled as 'none of the above', which did not fall neither under any of the 6 Ekman labels nor 'neutral'. The second type of removed frames were asociated with more than one emotion, and their inclusion would change the ML task to a multi-label classification problem (Huang et al., 2019; Durand et al., 2019). Removing these frames resulted into 2,531 facial images annotated with the 6 Ekman emotions plus 'neutral'.
The second preprocessing step consists of applying a

emotion	data distribution
Anger	18.30%
Disgust	7.72%
Fear	12.43%
Happy	7.92%
Neutral	7.58%
Sad	14.36%
Surprise	31.85%

Table 1: Labels distribution in the training set.

abbr.	technique
FT	Fine-tuning
SGD	Stochastic Gradient Descent optimizer
SAP	Sharpness Aware Minimization
IP	Image preprocessing
HF	Horizontal flip
CW	Class weights

Table 2: Abbreviations used for the techniques used during the experiments

face cropping (Section 3.1) to the images before feeding them to the CNN. This was needed because the images in the FePh dataset include some parts of the upper body. This conforms with the pre-processing applied to the pre-trained models. The data distribution across the different emotion classes in the training set is shown in table 1.

Finally, the FePh sign-language dataset was randomly split in a training (80%, used for fine-tuning the pre-trained models) and a test set (20%, used for evaluation). Images belonging to the same video sequence were kept in the same split. This phase allowed trying 10 configurations on top of the baselines.

The small size of the dataset raises questions on whether the results may generalize in a bigger dataset. For this purpose, we applied a **5-fold cross-validation** test to the two baselines models and to their 4 most promising varied configurations. Across the 5 folds, the average accuracy and sensitivity per class were calculated together with their standard deviation.

4.3. Pre-trained Models for Facial Expression Recognition (FER)

Here we provide details about the pre-trained models available for FER, which aim to recognize the seven basic Ekman emotions on generic datasets. As explained, these pre-trained models were fine-tuned on the SL-specific dataset and several techniques were added. To get the state-of-the-art results, the models presented by Savchenko (2021), which provide a lightweight CNN for the recognition of facial emotions based on two different architectures, were chosen as they achieve state-of-the-art accuracy.

The two models that were further used in the experiments are based on two CNN architectures: (1) MobileNet and (2) EfficientNet (Section 3.2). Both pre-trained models were trained on the AffectNet dataset (Mollahosseini et al., 2017) which includes almost 440k annotated images, having before been pre-trained on the much larger VGGFace2 dataset (Gennaro and Vairo, 2019).

The abbreviations used for the techniques used during the experiments are shown in table 2. All of the experiment configurations are summarized in table 3 and are detailed in the following two sections for the exploratory and the cross-validation phase, respectively.

4.4. Exploratory Phase

This exploration consists of a combination between pre-processing techniques and hyperparameters that were tested on a single 20% FePh data split.

4.4.1. Experiments with MobileNet

No-FT: Baseline Pre-trained Model This experiment was performed to check how the existing pre-trained model performs when tested on the SL data.

FT: Simple Fine-tuning This configuration consists of fine-tuning the pre-trained model with the 80% split of the FePh. A simple fine-tuning approach was used for the MobileNet architecture. In a CNN, the last layer learns the high-level features, and hence the last few layers are sufficient for transfer learning (Tajbakhsh et al., 2016). The last layer of the pre-trained model was first removed and a new dense layer was added to the CNN and all the previous layers of the base net were frozen to train just the last layer. This last layer was then trained on the new dataset including images from the SL (FePh) dataset for 3 epochs. Finally, all the previous frozen layers were unfrozen and the entire CNN was trained on the FePh data for 7 more epochs. The categorical cross-entropy loss was optimized by the Adam optimizer with a learning rate equal to 0.001.

FT-SGD: Fine-tuning with Stochastic Gradient Descent (SGD) In this configuration, following Savchenko (2021), the baseline model was fine-tuned with Adam optimizer for 5 epochs and SGD was used for the last two epochs with learning rate of 0.0001.

FT-SGD + CW: Class Weights for an Imbalanced Fine-tuning Set As explained (section 3.3), CNNs may perform poorly because of an imbalance in the fine-tuning data caused by a significant difference in amount of data in a class compared to the others, resulting in an insufficient representation of the minority classes. To tackle this imbalance across classes, we assign different class weights to each of the classes in the training data. This results in increasing the loss value for the classes that are insufficiently represented. The data distribution across classes in the fine-tuning dataset is shown in table 1.

FT-SGD + HF: Fine-tuning with Data Augmentation We horizontally flip images before feeding them to the CNN, thus doubling the training data.

architecture	configurations		description
	exploratory	c/v	
MobileNet	No-FT	M0	Base model
	FT		Simple fine-tuning of base model with Adam optimizer
	FT-SGD		Fine-tuning of base model with Adam and SGD optimizers
	FT-SGD + CW		FT-SGD + Classes balanced with class weights
	FT-SGD + HF		FT-SGD + Training dataset augmented with images flipped horizontally
	FT-SGD + IP		FT-SGD + Images normalized before training
	FT-SGD + IP + HF + CW	M1	FT-SGD + Image normalization, horizontal flip and class weights
	FT-SGD + IP + HF	M2	FT-SGD + image normalization and horizontal flip
EficientNet	No-FT	E0	Base model
	FT + SAM + HF + CW	E1	Base model fine-tuned with SAM optimizer + horizontal flip + class weights
	FT + SAM + HF	E2	Base model fine-tuned with SAM optimizer + horizontal flip
	FT-SGD + SAM + HF		Base model fine-tuned with SAM and SGD optimizers + horizontal flip

Table 3: Configurations used for the experiments

FT-SGD + IP: Fine-tuning with Image Preprocessing Along with fine-tuning, the images were normalized using the preprocessing function in Keras, where each color channel is zero-centered with respect to the ImageNet dataset (Ketkar, 2017; Savchenko, 2021).

FT-SGD + IP + HF + CW This is a direct replication of the similar experiment performed by Savchenko (2021), but by fine-tuning the pre-trained model with the FePh fine-tuning dataset. Since it had provided a state-of-the-art accuracy, this setting was tested to see if the combined effects of fine-tuning with data augmentation, image preprocessing, and class weights would improve the accuracy also with the FePh dataset compared to the previous settings.

FT-SGD + IP + HF Fine-tuning with data augmentation and image preprocessing. Here, the experiment was repeated with image preprocessing and horizontal flipping but without class weights.

4.4.2. Experiments with EfficientNet

As discussed in Chapter 3, the EfficientNet architecture provides better accuracy on ImageNet than MobileNet, and is considered a powerful tool in CV (Wang and Yu, 2021). Hence, the MobileNet experiments with the higher accuracy were replicated for EfficientNet to allow comparison of the two architectures. Among the EfficientNet variants, the EfficientNet-B0 architecture was chosen, as its default input image size (224x224) is the same as the size of the images in the dataset. Image preprocessing (IP) was not considered for this architecture as it was not suggested for the base models of Savchenko (2021).

No-FT: Pre-trained Model The baseline EfficientNet configuration pre-trained on AffectNet data and tested on the FePh test set.

FT + SAM + HF + CW: Fine-tuning with Sharpness Aware Minimization, Data Augmentation, and Class Weights This experiment is a replica of the pre-trained model provided by Savchenko (2021), but with additional fine-tuning on the FePh dataset. Moreover, this configuration uses the Sharpness Aware Minimization (SAM) optimizer (Foret et al., 2020). Initially, only the last layer is fine-tuned on FePh, for 3 epochs, with a learning rate of 0.001 while freezing all layers in the base net. Finally, all the layers are trained with the SAM optimizer with a learning rate of 0.0001 for 6 epochs as was proposed by Savchenko (2021). This experiment has also used horizontal flip as data augmentation technique and class weighting.

FT + SAM + HF: Fine-tuning with SAM and Data Augmentation This configuration uses fine-tuning with SAM along with horizontal flip. Class weighting was removed to check its contribution.

FT-SGD + SAM + HF: Fine-tuning with SAM and Data Augmentation and SGD This configuration tests the results using Stochastic Gradient Descent as optimizer while fine-tuning, because it was found to give the best results with MobileNet.

4.5. Cross-validation Phase

The best performing configurations from the exploratory phase were further evaluated by performing cross-validation (summarized in table 3 with their abbrevations shown in the ‘c/v’ column). Their description follows.

No-FT (M0 & E0) As a baseline, two pre-trained models (one for MobileNet and one for EfficientNet) presented by Savchenko (2021) were evaluated each on the 5 test sets obtained after splitting the FePh data into 5 folds. The accuracy and sensitivity per class were calculated and then the average and standard deviation was calculated.
The MobileNet configurations chosen for the cross-validation phase were:

FT-SGD + IP + HF + CW (M1) providing a state-of-the-art accuracy on AffectNet (Savchenko, 2021) and

FT-SGD + IP + HF (M2) showing promising results during the exploratory phase, despite the lack of class weighting, so it was chosen for cross-validation.

Cross-validation was also performed on the Efficient-Net architecture with the respective configurations **FT + SAM + HF + CW (E1)** and **FT + SAM + HF (E2)**, which have been described in the previous section.

5. Results

This section showcases the results obtained from experiments conducted with two CNN architectures (MobileNet-v1, EfficientNet-B0) with several data processing techniques in both the exploratory phase the cross-validation phase. As **evaluation metrics** the model accuracy, the sensitivity per class and the average sensitivity are given for every model. As **baseline** we consider the result obtained from the experiment conducted without fine-tuning, and the rest of the experiments are compared against that.

5.1. Exploratory Phase: MobileNet-v1

As shown in table 4, fine-tuning improved the overall accuracy for more than 13%, whereas there was an additional small improvement when the model was optimized with the Adam optimizer for the first few epochs and with the SGD optimizer for the last 3 epochs. It can be also seen that adding class weights alone reduced the overall accuracy, but when class weights were combined with image preprocessing and horizontal flip, it provided the highest average sensitivity of 63.3%. Data augmentation with horizontal flipping did not provide any improvement in the accuracy of the fine-tuned model. Similarly, image normalization did not improve the accuracy. However, when data augmentation was combined with image normalization, the accuracy was increased to 67% providing the best accuracy across all the models trained.

The setting with the best overall accuracy (M0) and the one with the best average sensitivity (M1) are chosen to be further investigated in the cross-validation phase.

5.2. Exploratory Phase: EfficientNet-B0

Table 4 shows that the best accuracy was provided by the configuration wich combines fine tuning with SAM and horizontal flipping. Similar to the MobileNet-v1 architecture, with class weighting from the base model, the accuracy is 2.6% higher. The EfficientNet-B0 models took 25% longer to fine-tune due to SAM as the main optimizer, as compared to the MobileNet-v1 models which use Adam for the most epochs and SGD for the last two epochs (section 3.2).

5.3. Cross-validation Phase

Table 5 shows the results obtained after averaging the accuracies across 5 models trained while performing 5-fold cross-validation, where the average sensitivity and the sensitivity per class were also recorded in a similar way. Since these metrics are averaged on different folds of training sets from the FePh dataset, they are more suitable in drawing overall conclusions, as the ones shown in the exploratory phase.

First, it can be observed that fine-tuning using the best combination of techniques outperforms the model with no fine-tuning (at least by 17%). This **confirms the main hypothesis** that for sign-language FER, fine-tuning a generic pre-trained model on a sign-language-specific dataset helps to improve the performance on this task. Additionally, it should be noted that the achieved overall accuracy of 62.4-62.8% is comparable to the state-of-the-art accuracy of the base models in the generic FER tasks (Savchenko, 2021).

For MobileNet-v1, the configuration that significantly gave the best accuracy was FT-SGD + IP + HF (M2). By comparing this configuration with its variant lacking class-weights, we can see that **class-weights are harmful** in this setting.

For EfficientNet-B0, it was found that the configuration FT + SAM + IP + HF (E2) gave the highest accuracy (62.8%). The difference of this setting with the configuration including class weights is not significant in this case, so one cannot say with confidence whether class weights are improving or harming EfficientNet.

No specific conclusion can be drawn regarding the differences between the MobileNet and EfficientNet architectures **both architectures seem equally suitable for the purpose of fine-tuning**. Nevertheless, one should consider that EfficientNet took slightly longer time to fine-tune, which might be an issue in future works, if larger fine-tuning datasets are considered.

The results based on the average sensitivity as well as the sensitivity per class vary a lot, e.g. one can see that different classes seem to be predicted best with different configurations. Nevertheless, these small sensitivity differences do not allow significant comparisons due to the very large standard deviations, which are attributed to the very small dataset. On the other side, we can confirm that the settings that provide significant accuracy improvements are still the optimal ones, since they do not cause a significant deterioration of the class and average sensitivities.

By comparing the 7 classes, we see that 'fear' has the lowest sensitivity (29.2%), followed by 'disgust' and 'neutral'. The best predicted class is 'happy' (82.7%) followed by 'surprise'.

6. Conclusion and Future Work

Through our experiments, we confirmed the hypothesis that fine-tuning a neural network alreaddy pre-trained to recognize facial expressions, togather with other the preprocessing techniques and optimizers, improves the model performance in classifying facial expressions on a sign language dataset. The achieved accuracy is satisfactorily high, as it is comparable to the state-of-the-art accuracy of the base models in generic FER of prior work. No significant difference was observed between the best configurations of MobileNet and EfficientNet architectures, but the training time for the EfficientNet models was higher than that of MobileNet.

The overall accuracy improved when image normaliza-

	configuration	c/v	acc.	sensitivity per class							avg.
				anger	disgust	fear	happy	neutral	sadness	surprise	sens.
MobileNet	No-FT	M0	52.0	54.5	**74.2**	26.3	15.8	26.2	11.9	79.6	41.0
	FT		65.4	81.1	38.7	45.6	63.1	35.7	50.8	77.8	56.1
	FT-SGD		65.7	**82.6**	58.0	47.4	73.7	33.3	57.6	70.0	60.4
	FT-SGD + CW		54.0	65.2	71.0	**56.1**	78.9	28.6	**61.0**	42.5	57.7
	FT-SGD + HF		64.7	77.3	51.6	50.9	63.2	28.6	55.9	74.3	57.4
	FT-SGD + IP		65.3	**82.6**	35.5	43.9	68.4	28.6	45.8	**80.2**	55.0
	FT-SGD + IP + HF + CW	M1	63.7	75.0	**74.2**	**56.1**	**84.2**	**38.1**	52.5	63.5	**63.3**
	FT-SGD + IP + HF	M2	**67.0**	81.8	61.3	40.4	68.4	33.3	50.8	79.6	59.3
EfficientNet	No-FT	E0	53.5	50.8	67.7	**36.8**	47.4	47.6	18.6	73.1	49.0
	FT + SAM + HF + CW	E1	63.9	62.9	67.7	**36.8**	84.2	**76.2**	66.1	67.1	65.9
	FT + SAM + HF	E2	**66.5**	**68.2**	67.7	31.6	84.2	69.0	**67.8**	**73.7**	**66.1**
	FT-SGD + SAM + HF		63.9	65.2	**71.0**	33.3	**89.5**	59.5	59.3	71.9	64.1

Table 4: The overall accuracy, sensitivity per class, and average sensitivity (in %) obtained for all configurations (described in table 3) of the exploratory phase (i.e., tested on a single fold). The configurations that have an abbreviation in the column 'c/v' are repeated later in the cross-validation phase (table 5).

	acc. (std)	average sensitivity per class (std)							avg.
		anger	disgust	fear	happy	neutral	sadness	surprise	sens. (std)
M0	44.1 (4.7)	44.4 (10.3)	**58.4 (2.4)**	**25.8 (12.4)**	47.3 (19.3)	20.8 (5.7)	17.2 (9.1)	63.8 (6.7)	39.7 (9.4)
M1	51.7 (7.4)	51.3 (21.4)	37.3 (13.8)	11.6 (5.5)	**56.6 (16.4)**	**75.3 (16.7)**	**60.4 (23.5)**	58.2 (12.0)	50.0 (15.6)
M2	**62.4 (3.2)**	**73.7 (5.9)**	41.9 (14.1)	23.6 (9.6)	53.7 (12.2)	50.2 (22.3)	57.2 (16.1)	**82.1 (6.7)**	**54.6 (12.4)**
E0	45.7 (4.7)	42.3 (7.7)	55.6 (3.9)	30.1 (14.4)	53.3 (12.7)	54.6 (16.8)	19.5 (11.3)	59.7 (12.1)	45.0 (11.3)
E1	62.2 (2.4)	**65.3 (9.3)**	**57.5 (12.6)**	**35.8 (13.3)**	72.3 (9.8)	**66.1 (10.0)**	**59.1 (16.8)**	68.5 (6.4)	**60.7 (11.2)**
E2	**62.8 (4.7)**	62.3 (8.6)	45.4 (18.8)	29.2 (16.6)	**82.7 (11.7)**	59 (17.6)	52.7 (22.2)	**79.2 (8.5)**	58.6 (14.9)

Table 5: The overall accuracy, sensitivity per class, and average sensitivity (in %) with the standard deviation (std) obtained for all the MobileNet-v1 and EfficientNet-B0 configurations calculated during the cross-validation phase

tion was used in combination with augmenting training data with horizontally flipped images. Despite the obvious lack of balance between the classes in the dataset, balancing classes with the help of class weights can harm the accuracy.

The best models obtained from the experiments conducted were configured as following: MobileNet-v1 fine-tuned with Stochastic Gradient Descent using tuning data normalized and augmented using horizontally flipped images, and EfficientNet-B0 fine-tuned with Sharpness Aware Minimization using tuning data augmented with horizontally flipped images.

It is obvious through our analysis that further work would require bigger datasets that would allow more robust results, if possible also including spatial and manual elements and offerring better resolution, a broader domain and reacher modalities (e.g., full video sequences). Additionally, one should consider coverage of other sign languages and cultural backgrounds, whereas we are actively working on the adaptation of the labels to include linguistic content/markers and other affective aspects relevant to communication purposes.

7. Acknowledgements

The work was accomplished as a MSc thesis, part of the program ICT Innovation (Technical University of Berlin). The supervision was funded by the project SocialWear (German Ministry of Research and Education; BMBF) and by the EU Horizon 2020 programme within the EASIER project (Grant agreement ID: 101016982).

8. Bibliographical References

Akhand, M., Roy, S., Siddique, N., Kamal, M. A. S., and Shimamura, T. (2021). Facial emotion recognition using transfer learning in the deep cnn. *Electronics*, 10(9):1036.

Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6. Ieee.

Atila, Ü., Uçar, M., Akyol, K., and Uçar, E. (2021). Plant leaf disease classification using efficientnet deep learning model. *Ecological Informatics*, 61:101182.

Boyko, N., Basystiuk, O., and Shakhovska, N. (2018). Performance evaluation and comparison of software for face recognition, based on dlib and opencv library. In *2018 IEEE Second International Confer-*

ence on Data Stream Mining & Processing (DSMP), pages 478–482. IEEE.

Chu, W.-S., De la Torre, F., and Cohn, J. F. (2017). Learning spatial and temporal cues for multi-label facial action unit detection. In *12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 25–32. IEEE.

Durand, T., Mehrasa, N., and Mori, G. (2019). Learning a deep convnet for multi-label classification with partial labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 647–657.

Ekman, P. (1999). Basic emotions. *Handbook of cognition and emotion*, 98(45-60):16.

Emami, S. and Suciu, V. P. (2012). Facial recognition using opencv. *Journal of Mobile, Embedded and Distributed Systems*, 4(1):38–43.

Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. (2020). Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*.

Gennaro, C. and Vairo, C. (2019). Improving multiscale face recognition using vggface2. In *New Trends in Image Analysis and Processing–ICIAP 2019: ICIAP International Workshops, BioFor, PatReCH, e-BADLE, DeepRetail, and Industrial Session, Trento, Italy, September 9–10, 2019, Revised Selected Papers*, volume 11808, page 21. Springer Nature.

Ghosh, T., Abedin, M. M.-H.-Z., Chowdhury, S. M., Tasnim, Z., Karim, T., Reza, S. S., Saika, S., and Yousuf, M. A. (2020). Bangla handwritten character recognition using mobilenet v1 architecture. *Bulletin of Electrical Engineering and Informatics*, 9(6):2547–2554.

Heidari, M., Mirniaharikandehei, S., Khuzani, A. Z., Danala, G., Qiu, Y., and Zheng, B. (2020). Improving the performance of cnn to predict the likelihood of covid-19 using chest x-ray images with preprocessing algorithms. *International journal of medical informatics*, 144:104284.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Huang, J., Qin, F., Zheng, X., Cheng, Z., Yuan, Z., Zhang, W., and Huang, Q. (2019). Improving multi-label classification with missing labels by learning label-specific features. *Information Sciences*, 492:124–146.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Johnson, J. M. and Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):1–54.

Kandel, I. and Castelli, M. (2020). The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT express*, 6(4):312–315.

Ketkar, N. (2017). Introduction to keras. In *Deep learning with Python*, pages 97–111. Springer.

Kim, D. H., Baddar, W. J., Jang, J., and Ro, Y. M. (2019). Multi-objective based spatio-temporal feature representation learning robust to expression intensity variations for facial expression recognition. *IEEE Transactions on Affective Computing*, 10(2):223–236.

Ko, B. C. (2018). A brief review of facial emotion recognition based on visual information. *sensors*, 18(2):401.

Koo, K.-M. and Cha, E.-Y. (2017). Image recognition performance enhancements using image normalization. *Human-centric Computing and Information Sciences*, 7(1):1–11.

Kossaifi, J., Tzimiropoulos, G., Todorovic, S., and Pantic, M. (2017). Afew-va database for valence and arousal estimation in-the-wild. *Image Vision Comput.*, 65(C):23–36, sep.

Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., and Matthews, I. (2010). The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, pages 94–101.

Marques, G., Agarwal, D., and de la Torre Díez, I. (2020). Automated medical diagnosis of covid-19 through efficientnet convolutional neural network. *Applied soft computing*, 96:106691.

Meng, D., Peng, X., Wang, K., and Qiao, Y. (2019). Frame attention networks for facial expression recognition in videos. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3866–3870. IEEE.

Michele, A., Colin, V., and Santika, D. D. (2019). Mobilenet convolutional neural networks and support vector machines for palmprint recognition. *Procedia Computer Science*, 157:110–117.

Mollahosseini, A., Hasani, B., and Mahoor, M. H. (2017). Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1):18–31.

Mollahosseini, A., Hasani, B., and Mahoor, M. H. (2019). Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing*, 10(1):18–31.

Ngo, Q. T. and Yoon, S. (2020). Facial expression recognition based on weighted-cluster loss and deep transfer learning using a highly imbalanced dataset. *Sensors*, 20(9):2639.

O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G. V., Krpalkova, L., Riordan, D., and Walsh, J. (2019). Deep learning vs. traditional computer vision. In *Science and Information Conference*, pages 128–144. Springer.

Pal, K. K. and Sudeep, K. (2016). Preprocessing for image classification by convolutional neural networks. In *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pages 1778–1781. IEEE.

Pitaloka, D. A., Wulandari, A., Basaruddin, T., and Liliana, D. Y. (2017). Enhancing cnn with preprocessing stage in automatic emotion recognition. *Procedia computer science*, 116:523–529.

Radenović, F., Tolias, G., and Chum, O. (2018). Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668.

Ravi, R., Yadhukrishna, S., et al. (2020). A face expression recognition using cnn & lbp. In *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pages 684–689. IEEE.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520.

Sauter, D. A., Eisner, F., Ekman, P., and Scott, S. K. (2010). Cross-cultural recognition of basic emotions through nonverbal emotional vocalizations. *Proceedings of the National Academy of Sciences*, 107(6):2408–2412.

Savchenko, A. V. (2021). Facial expression and attributes recognition based on multi-task learning of lightweight neural networks. In *2021 IEEE 19th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 119–124.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48.

Sinha, D. and El-Sharkawy, M. (2019). Thin mobilenet: An enhanced mobilenet architecture. In *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0280–0285. IEEE.

Soo, S. (2014). Object detection using haar-cascade classifier. *Institute of Computer Science, University of Tartu*, 2(3):1–12.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., and Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299–1312.

Tan, M. and Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR.

Tan, M. and Le, Q. (2021). Efficientnetv2: Smaller models and faster training. In *International Conference on Machine Learning*, pages 10096–10106. PMLR.

Taqi, A. M., Awad, A., Al-Azzo, F., and Milanova, M. (2018). The impact of multi-optimizers and data augmentation on tensorflow convolutional neural network performance. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 140–145. IEEE.

Wang, K. and Yu, X. (2021). Mobilenet and efficientnet demonstration on google landmark recognition dataset. *International Core Journal of Engineering*, 7(3):313–319.

Xiang, J. and Zhu, G. (2017). Joint face detection and facial expression recognition with mtcnn. In *2017 4th international conference on information science and control engineering (ICISCE)*, pages 424–427. IEEE.

Zheng, Q., Yang, M., Tian, X., Jiang, N., and Wang, D. (2020). A full stage data augmentation method in deep convolutional neural network for natural image classification. *Discrete Dynamics in Nature and Society*, 2020.

9. Language Resource References

Alaghband, M., Yousefi, N., and Garibay, I. (2021). Facial expression phoenix (feph): An annotated sequenced dataset for facial and emotion-specified expressions in sign language. *International Journal of Electronics and Communication Engineering*, 15(3):131–138.