EMOTION RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK

Franko Šikić
University of Zagreb
Zagreb, Croatia
franko.sikic@fer.hr

Marko Šandrk
University of Zagreb
Zagreb, Croatia
marko.sandrk@fer.hr

Jinyuan Li
North China Electric Power University
Baoding, China
201709000712@ncepu.edu.cn

Emilia Szymańska
Wroclaw University of Science and Technology
Wroclaw, Poland
248975@student.pwr.edu.pl

Igor Zieliński
Wroclaw University of Science and Technology
Wroclaw, Poland
248944@student.pwr.edu.pl

***Abstract*—This paper reviews the tools and baseline models used in the team project conducted at University of Zagreb. Project's goal was to train the Convolutional Neural Network to recognize human emotions from face images. Pre-trained MobileNetV2 and EfficientNetB0 were used as baseline models, while the optimal architectures and hyper-parameters were found with the help of 4-fold cross validation. Obtained results are presented alongside with performance comparison of used models.**

***Index Terms*—CNN, MobileNet, EfficientNet, emotion recognition, image analysis.**

I. Introduction

Emotions play a vital role in any inter-personal communication. Therefore, it is unsurprising that the recognition of facial emotion has been a crucial subject of much recent research. There has been interest in human emotion recognition in many different fields including, but not limited to, human-computer interface [1], safe driving [2], online education [3], animation [4] and security [5].

Emotions are expressed in various ways, such as facial expression, voices, physiological signals and text [6]. Among these features, facial expressions are the most useful ones since they are visible, they contain many useful information for emotion recognition, and it is easier to collect a large dataset of faces (than other means for human recognition) [7].

In the last 10 years, deep learning, as a part of a broader family of machine learning methods based on artificial neural networks with representation learning, has been growing rapidly since these methods can be used in various fields of science and technology. Especially, convolutional neural networks (CNNs), a class of deep neural networks, have made great success in image recognition [8]. They can achieve good performance in a variety of image classification tasks.

In this study, we took advantages of MobileNetV2 and EfficientNetB0. Explanation of the key characteristics of these models are in the Related work section. We used EfficientNetB0 as it is a part of EfficientNet family which has been providing the best results in image classification problems for the last couple of years. MobileNetV2 was chosen too as it is a very small model with good performance. Both models were pre-trained on ImageNet dataset as we wanted to take advantage of transfer learning. Used dataset is ExpW Cleaned dataset [9], which contains a total of 84,830 pictures, with 7 labels. The distribution of our dataset is shown in Table I, while examples of images are presented in Fig. 1.

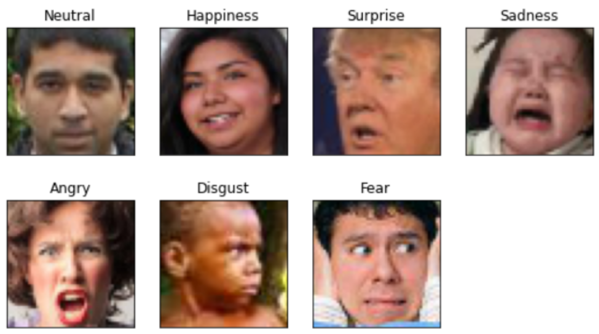

Fig. 1. Examples of images.

Dataset Distribution	
Emotion	Samples
neutral	31188
happiness	28111
sadness	10393
surprise	6848
disgust	3688
anger	3548
fear	1047

TABLE I
Data Distribution

II. Related work

A. MobileNetV1

Although a standard Convolutional Neural Network (CNN) is more efficient than a traditional multilayer perceptron, it still demands a lot of parameters as well as computational expences. To reduce both model size and complexity, a Depthwise Separable Convolution approach has been introduced by Google Inc. researchers [10]. MobileNetV1's principle of operation is based on the two following steps:

- depthwise convolution,
- pointwise convolution.

As shown in Fig. 2[1], depthwise convolution is the channel-wise $D_K \times D_K$ spatial convolution, while the next step is in fact 1×1 convolution performed to change the dimension.

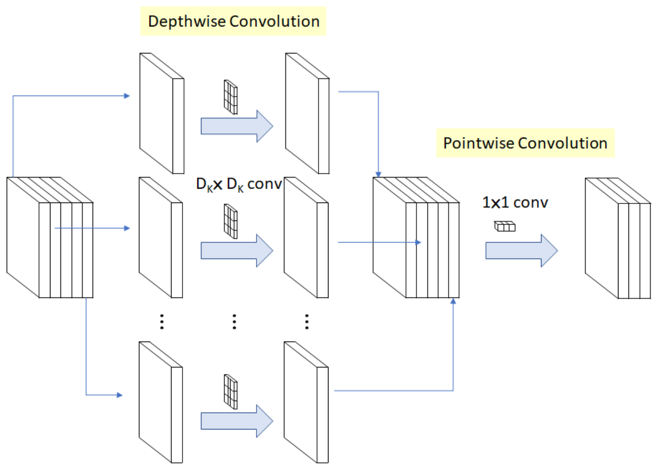

Fig. 2. Depthwise Separable Convolution.

Due to the use of this method, a significant computation reduction is achieved. In comparison with standard convolutional network, the cost ratio of depthwise separable convolution is equal to $\frac{1}{N} + \frac{1}{D_K^2}$, where N is the number of output channels and D_KxD_K stands for the kernel size.

In MobileNet, Batch Normalization (BN) and ReLU are applied after each convolution. Additionally, two new parameters are introduced for the MobileNet to be regulated easily: Width Multiplier α and Resolution Multiplier ρ. Width Multiplier's purpose is the control of a layer's input width, while Resolution Multiplier is in charge of scaling the input image resolution of the network.

The researchers in [10] have proved that MobileNet approach results in much fewer multiplication and additions (Mult-Adds) alongside with parameters with only a slight loss in accuracy. It has been presented with an example of ImageNet dataset shown in table II.

[1]Source: https://towardsdatascience.com/review-mobilenetv1-depthwise-separable-convolution-light-weight-model-a382df364b69

Model	Accuracy	Million Mult-Adds	Million Parameters
Standard Conv.	71.7%	4866	29.3
Depthwise Separable Conv.	70.6%	569	4.2

TABLE II
Comparison of two approaches with ImageNet dataset

Some of the applications of MobileNet are object detections, finegrain classification, landmark recognition and face attributes, therefore this architecture of neural network has been used in the emotion recognition.

B. MobileNetV2

MobileNetV2 presented in [11] is a successor of MobileNetV1 - its architecture as well uses the idea of Depthwise Separable Convolution, but it expands it with two other approaches: inverted residuals and linear bottlenecks.

A residual block consists of a convolutional block whose first and last layer are connected with a skip connection. This approach gives the neural network a possibility to access activations that were not modified in the convolutional block earlier. The input has a high number of channels, which are then compressed with 1x1 convolution. In order to add input and output in the end, the number of channels is increased again, as it is shown in Fig. 3 (source: [11]).

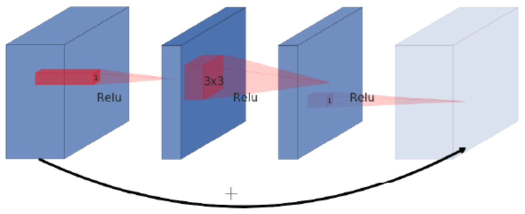

Fig. 3. Residual block.

MobileNetV2 follows an inverted approach. The first step is to widen the network using a 1x1 convolution. At the end, another 1x1 convolution squeezes the network in order to match the initial number of channels. The steps are shown in Fig. 4 (source: [11]).

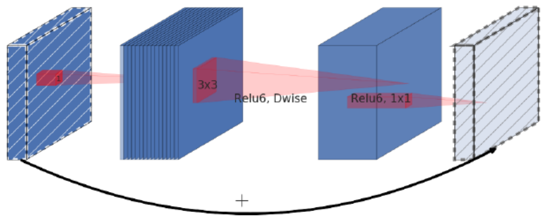

Fig. 4. Inverted residual block.

Because of this inversion, the block has far fewer parameters.

The idea of linear bottleneck is to add a linear output of the last convolution of a residual block to the initial activations. Therefore, the discard of values that are smaller than 0 and hence the loss of information is prevented.

As for the non-linear activation functions, the researches of [11] use ReLU6 instead of ReLU. It limits the value of activations to a maximum of 6 - between 0 and 6 it is linear. It might be helpful when it is needed to limit the information left of the decimal point to 3 bits.

C. *EfficientNet*

From 2012 on, CNNs have been widely used for a variety of tasks in deep learning, especially in the field of computer vision. Due to their wide usage, researchers have been trying to come up with architectures that can improve accuracy of the model on different tasks. EfficientNet achieves better accuracy and efficiency using model scaling.

In context of CNNs scaling can be interpreted with respect to three factors:

- depth - number of layers,
- width - number of channels in Conv layer,
- resolution - image resolution that is being passed to a CNN.

If increasing one of scaling factors mentioned above increases model's accuracy, the effect isn't linear and effect saturates quickly as shown of Fig. 5 (source: [12]).

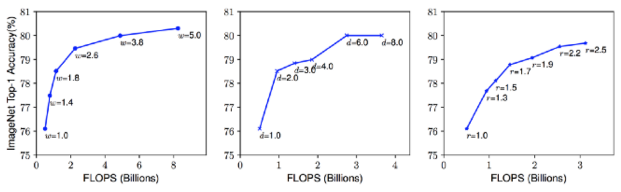

Fig. 5. Scaling Up a Baseline Model with Different Network Width (w), Depth (d), and Resolution (r) Coefficients. Bigger networks with larger width, depth, or resolution tend to achieve higher accuracy, but the accuracy gain quickly saturate after reaching 80%, demonstrating the limitation of single dimension scaling.

Better accuracy can be achieved using combined scaling that changes more than one scaling factor. Researchers of [11] suggested a method called Compound Scaling.

$$\begin{aligned}
\text{depth: } & d = \alpha^{\phi} \\
\text{width: } & w = \beta^{\phi} \\
\text{resolution: } & r = \gamma^{\phi} \\
\text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\
& \alpha \geq 1, \beta \geq 1, \gamma \geq 1
\end{aligned}$$

Fig. 6. Compound Scaling method

Value Φ is coefficient that is proportional to how many resources are available, while α, β and γ tell how to distribute those resources along network depth, width, and resolution scaling.

After choosing baseline model those parameters can be found in two steps procedure:

1) fix $\Phi = 1$ and do a small grid search for α, β, and γ,
2) fix α, β, and γ as constants (with values found in above step) and change the values of Φ; the different values of Φ produce EfficientNets B1-B7($\Phi = 2$ produces EfficienNetB2 etc.).

III. Our work

Before approaching the training of our models, dataset was transferred to TFRecords [13] which allowed to exploit benefits of Tensor Processing Units (TPUs) [14]. TPU is an application-specific integrated circuit (ASIC), developed by Google Inc., used to accelerate linear algebra operations characteristic for machine learning. Relying on TPUs meant that TensorFlow library [15] would be used in developing of our models.

Dataset was divided into training subset and testing subset in 80:20 ratio. To achieve better accuracy as well as stability and generalization of our models, 4-fold cross validation was applied. In that way all our examples in the training dataset were used both in training of our models and in the validation of our models. Also, by using this approach, results which were obtained from experimenting with various values of different hyper-parameters were comparable and we were able to unambiguously determine which values of certain hyper-parameters benefit our models accuracy. During the training of the models, categorical cross-entropy loss function was used as well as a learning rate scheduler to assure convergence during the learning process. To boost models' generalization capabilities, augmentation was used. More precisely, rotation, shear, zoom and shift were each applied on the images during the training process. Furthermore, flipping the images left to right horizontally was also performed.

Two different models were trained. One model had pretrained MobileNetV2 as its baseline model and the other one used pretrained EfficientNetB0 as its baseline model.

IV. Results

As it was previously mentioned, we have trained two different models which used different pretrained models as their baseline models - one with MobileNetV2 model and the other with EfficientNetB0 model.

First, we trained basic models (row one of Table III) which consisted of fully convolutional layers of baseline models and a global average pooling layer (GAP) followed by an output layer. After that, some basic augmentation (rotation, shift, shear, zoom and horizontal flip) was added, which increased the models' accuracy (Table III). Finally, after some experimenting, optimal architecture for the model (row three of Table III) with MobileNetV2 as baseline model proved to consist of fully convolutional layers of MobileNetV2 followed by a GAP layer, a fully connected layer of neurons with sigmoid activation functions, a dropout layer (so as to prevent overfitting) and an output layer with softmax activation function. Similarly, for the second model, optimal architecture proved to consist of a GAP layer appended to the baseline model followed by two fully connected layers with sigmoid activation function and an output layer using softmax as its activation function. As for the first model, a dropout layer was added between each of the fully connected layers.

	Baseline model	
Type of model	MobileNetV2	EfficientNetB0
basic	63.3%	66.8%
basic + basic aug	70.2%	70.1%
optimized	71.2%	71.7%

TABLE III
4-FOLD CROSS VALIDATION ACCURACY RESULTS

After determining optimal architectures, search for various hyper-parameters was conducted to determine optimal batch size (the batch size of 64 provided best results), optimal parameters of learning rate scheduler as well as optimal parameters for each of the augmentation techniques applied to the model. It is worth mentioning that several different optimizers were used for training such as Stochastic Gradient Descent (SGD) and Nesterov accelerated gradient, but Adaptive Moment Estimation (Adam) resulted in the best outcome.

	Baseline model	
	MobileNetV2	EfficientNetB0
Accuracy	71.2%	71.7%

TABLE IV
ACCURACY ON THE TEST DATASET

As Table IV shows, both optimal models achieved same accuracy when tested on the examples from the testing dataset. This indicates that using 4-fold cross validation while training the models provided stability.

Table V and Table VI show confusion matrices for the model with MobileNetV2 as its baseline model and the model with the EffiecientNetB0 as its baseline model respectively. These tables clearly show that our models were not able to properly classify two emotions - disgust and fear.

		Predicted emotions					
		neutral	happiness	surprise	sadness	anger	Acc.
Actual emotions	neutral	5397	517	109	215	46	85.9%
	happiness	784	4620	91	72	15	82.8%
	surprise	328	108	760	68	91	56.1%
	sadness	871	141	64	900	71	44.0%
	anger	188	47	72	41	396	53.2%
	disgust	512	50	34	79	59	0%
	fear	24	14	96	36	38	0%

TABLE V
CONFUSION MATRIX FOR MOBILENETV2 MODEL

		Predicted emotions					
		neutral	happiness	surprise	sadness	anger	Acc.
Actual emotions	neutral	5416	517	123	198	30	86.2%
	happiness	725	4681	79	82	15	83.9%
	surprise	301	112	801	58	83	59.1%
	sadness	875	145	76	888	63	43.4%
	anger	192	46	86	38	382	51.3%
	disgust	539	56	36	64	48	0%
	fear	28	12	101	25	42	0%

TABLE VI
CONFUSION MATRIX FOR EFFICIENTNETB0 MODEL

V. DISCUSSION

At the beginning, the importance of applying augmentation to images has to be mentioned. By using augmentation the network is being fed with new images epoch after epoch, which enables network to learn more and generalize better. Adding just a little bit of augmentation improves the results around 3% (EfficientNetB0) and 7% (MobileNetV2). After many experiments, we found optimal hyper-parameters of both networks and results gained with those networks were 1% (MobileNetV2) and 1.6% (EfficientNetB0) better than the results of basic models with basic augmentation.

Results of the test phase were the same as the results of optimized models from 4-fold cross validation phase. This could mean that our K-fold cross validation strategy was quite stable, but another reason why we got these results could be that distribution of classes in the test set was very similar to the distribution of training data. It is also noticeable that MobileNetV2, even though it has almost two times less parameters than EfficientNetB0 (see Table VII and Table VIII), performs just slightly worse both in 4-fold cross validation and test phase.

Layer	No. of parameters
MobileNetV2	2,257,984
FC	40,992
Output	231
	Total: 2,299,207

TABLE VII
NUMBER OF PARAMETERS OF MOBILENETV2 MODEL

If we take a look at the confusion matrices in Table V and Table VI, we can see that we have obtained decent results, but there is one problem: both models are unable to predict disgust and fear. Considering the fact that these two classes are among 3 classes that are the least represented ones in the dataset, it is quite understandable that the model was not able to learn how to predict them. Both models are very accurate when it comes to predicting neutral and happiness (more than 80%), while surprise, sadness and anger are much harder to classify (around 50%). It is interesting to see that MobileNetV2 actually performs a bit better at classifying sadness and anger. Results show that neutral is most often being confused with happiness, while other emotions except fear are usually confused with neutral. This was expected as neutral and happiness are 2 most common classes in the dataset. On the other hand, fear is actually being confused with surprise quite often.

Layer	No. of parameters
EfficientNetB0	4,049564
1st FC	81,984
2nd FC	2,080
Output	231
	Total: 4,133,859

TABLE VIII
NUMBER OF PARAMETERS OF EFFICIENTNETB0 MODEL

VI. CONCLUSIONS

In this paper, we utilize two emotion recognition models based on MobileNetV2 model and EfficientNetB0 model. Augmentation proved to be very important for reaching better generalization as it boosted accuracy of basic models from 3% to 7%.

MobileNetV2 turns out to be a better solution as it provided almost the same result as EfficientNetB0, although it has twice less parameters. Actually, it performs even better than EfficientNetB0 at some particular classes (sadness and anger).

Class imbalance is a significant problem in this dataset. It is a main reason why two of the classes cannot be predicted by our models and future work on this project should tackle this problem. Several approaches can be used, such as using a custom loss function like weighted categorical cross-entropy or using Generative Adversarial Network (GAN) in order to generate more images of the classes which are least represented in the dataset.

REFERENCES

[1] Cowie, Roddy, Ellen Douglas-Cowie, Nicolas Tsapatsoulis, George Votsis, Stefanos Kollias, Winfried Fellenz, and John G. Taylor. ”Emotion recognition in human-computer interaction.” 2001

[2] Jeong, Mira, and Byoung Chul Ko. ”Driver’s Facial Expression Recognition in Real-Time for Safe Driving.” 2018

[3] Krithika L.B, Lakshmi Priya GG, Student Emotion Recognition System (SERS) for e-learning Improvement Based on Learner Concentration Metric 2016

[4] Aneja, Deepali, Alex Colburn, Gary Faigin, Linda Shapiro, and Barbara Mones. ”Modeling stylized character expressions via deep learning.” 2016

[5] Saste, Sonali T., and S. M. Jagdale. ”Emotion recognition from speech using MFCC and DWT for security system.” 2017

[6] El Ayadi, M.; Kamel, M.S.; Karray, F. Survey on speech emotion recognition: Features, classification schemes, and databases. 2011

[7] Carrier, P. L., Courville, A., Goodfellow, I. J., Mirza, M., & Bengio, Y. FER-2013 face database. Universit de Montreal, 2013.

[8] LeCun, Y. Generalization and network design strategies. Conectionism in perspective, 1989.

[9] Cleaned ExpW Dataset. Retrieved from https://github.com/markson14/ExpWCleaned

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications“, 2017

[11] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks“, 2018

[12] Mingxing Tan, Quoc V. Le EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks 2019

[13] TFRecord and tf.train.Example. Retrieved from https://www.tensorflow.org/tutorials/load_data/tfrecord

[14] Cloud Tensor Processing Units (TPUs). Retrieved from https://cloud.google.com/tpu/docs/tpus

[15] TensorFlow. Retrieved from https://www.tensorflow.org/