# Facial Image Emotion Recognition based on Cartoonized Features

Jingze Gong, Hanyang Feng, Zhiyuan Cao, Yue Mo [*]

## Abstract

*This paper presents a method on facial image emotion recognition by combining cartoonization and VGGNet. By applying cartoonization as preprocess to images, we can reduce unrelated facial features which often appear in traditional CNN method in facial emotion recognition task, such as wrinkles and freckles. A variant of VGGNet is used to to train on the cartoonized images, and saliency maps generated to prove cartoonization's effect.*

*The learning objectives of our method include total accuracy and separate accuracy of each emotion class so that each emotion can be classified well. Comparasions are also done between our model and traditional VGGNet and ResNet method to validate our model's effectiveness of our approach.*

*To the best of our knowledge, our model achieves top 3 total classification accuracy on FER2013 dataset in single network, and achieves best accuracy on surprise and disguist class in top 3 approaches. By examining saliency maps, we further prove that cartoonization has positive effect on emotion recognition by reducing unrelated features in facial images.*

## 1. Introduction

The ability to recognize users' emotions for the purpose of emotion engineering is currently one of the main endeavors of machine learning in affective computing. Emotion recognition task obtains emotion features and classify emotions based on facial expressions from images.Most CNN-based facial image emotion recognition works are carried out as it is feasible to perform simple classification on emotion dataset like FER2013, and also the evaluation of model performance is simple and straightforward [4] [12] [14] [15].

One of the most popular facial image emotion recognization datasets is FER2013. FER2013 was introduced at the 2013 International Conference on Machine Learning(ICML) and since then became a benchmark in evaluating model performance in emotion recognition. It is a specific emotion recognition dataset that contains complex naturalistic conditions and several different challenges. According to FER2013, human performance on the dataset is estimated to be about 65.5 % [5]. In comparing different methods with our results, we strictly consider previous work trained and evaluated on this dataset.

However, traditional CNN methods often attach too much importance on some features that are unrelated to facial expression. Skin details on face such as frackles and wrinkles have little relationship with emotion, but are extracted as features(Figure 1).

To address the above-mentioned problem, we expolre a new method by using cartoonization method, conbining with a VGGNet model structure to reduce unimportant features on facial images. We first construct the model structure as Figure 3, 4, then train model and imporve accuracy rate with different optimization algorithms, learning rate schedulers and fine-tuning methods. After that, saliency maps are then generated and analyzed to further explore the model's performance and the effect of cartoonization. Finally, we conclude with a discussion on our achievements from the performance of our model and efforts of cartoonization and prospective future research directions, which will be beneficial for expanding the body of knowledge in using cartoonization in the field of emotion recognition.

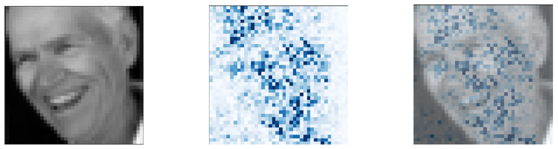
Figure 1. Real human face and saliency map from Yousif et al.'s VGGNet [7]. Most features exracted are on forhead, jaw and cheek wrinkles instead of facial features such as eyes and mouth.

## 2. Related Work

### 2.1. Facial Image Emotion Classification Method

Nowadays, CNN-based methods are widely applied in facial emotion recognization tasks. CNNs have shown great potential in image processing [9]. After the 2010s, the growth of computing power and the collection of larger datasets let CNNs be a much more feasible method in feature extraction

[*]This submission is the project proposal for course ECE4880J in 2022SU. This course is held by UM-SJTU Joint Institute, Shanghai Jiao Tong University. The mentor of this course is Dr. Siheng Chen.

and image classification [8].

Pecoraro et al.proposed a Local Multi-Head Channel Self-Attention(LHC) model, an attention-based ResNet approach for facial emotion recognition [12]. They split the input into several channels and then go through processing blocks such as convolution and max/average pooling, finally apply an attention layer and concatenate the channels together for classification. Their method achieves good result on accuracy, but need more data outside FER2013 dataset.

Georgescu et al. proposed a multi-model approach on FER2013 dataset [12]. In their work, they use VGGNet and bag-of-visual-words model to extract different features and concatenate them together for classification. Their model obtained 75.42% accuracy on FER2013 dataset, but also need additional training data like FER+.

Yousif and Chen use a single VGGNet structure and perform detailed optimization on hyperparamters [7]. Compared to other neural network approaches, VGGNet is simple in structure. They achieve state of the art accuracy 73.59% on Fer2013 dataset, and in all six classifications, "happiness" reaches accuracy of 89%.

## 2.2. Optimization Methods on CNN Structures

A great amount of works has also been done in using different optimization methods in CNN training. Research results show that an optimization algorithm that fits the model can effectively improve the model's performance [17]. Among all the works that result in good accuracy on FER2013, various optimization method are used. Georgescu used SVM for classification [12]. Pranav et al. used Adam in their DCNN [14]. Pecoraro et al. used SGD optimization in their LHC model [12].

Another important factors that affects the model performance is the learning rate. On one hand, if the learning rate is set too large, it could lead to model oscillations and not coverges to the minimum, or result in instability of the loss. On the other hand, if learning rate is set too small, the model's convergence will be slowed down and could be trapped at a local minimum which is not optimal. So it is important to use a learning rate scheduler to adjust the learning rate refering to a fixed rule during training [2]. Some scheduler refers to timebased decay, which reduces the learning rate by a factor calcualted by a linear or exponential function taking iteration number or epoch as input. Another type of schedule tries to automatically adjust model's learning rate based on the current local gradients during training process, such as Reducing Learning Rate on Plateau(RLRP), which decreases the learning rate when the accuracy does not increase for several epochs.

In Yousif and Chen's expreiment [7], they have compared the result accuracy based on five learning rate scheduler: ReduceLROnPlateau, StepLR, OneCycleLR, CosineAnnealingWarmRestarts and CosineAnnealingLR and five optimizer: SGD, ASGD, Adam, Adagrad and Adadelta. They found an obvious difference between different combinations. Within constant learning rate and same hyperparameters, the result on their model has 72% accuracy with SGD while has only about 69% accuracy with Adadelta. Besides, when they kept hyperparameters and optimizer as SGD unchanged and change learning rate scheduler, they found a 2% difference between Learning Rate on Plateau(RLRP) and OneCycleLR. After selecting the best learning rate scheduler and optimizer, they created saliency maps to clearly show the detail of the model, thus make the process of the model prediction clear. They also listed the result predict rate in a Confusion grid to illustrate the mispredict rate. Their expreiment procedure is well performed, and can be applied in our model.

## 2.3. Cartoonization Method

Wang et al. constructs a GAN system that can generate high-quality cartoonized images from real-world photos. [19]. The GAN system contains one generator and two discriminators. It first decomposes the image to obtain the surface representation, structure representation and texture representation of the image, then use these three representations to help network learn different features, where surface representation is the representation to imitate cartoon-like surface, structure representation is to perform cartoon-oriented segmentation, and texture representation is to generate images of different style.

In this system, the two descriminators are used to judge the similarity of surface and texture representations between generated cartoon image and the obtained standard representation. Within the descriminators, loss of these representations can be calculated. The structure representation loss can also be obtained by using L2 norm of the generated cartoon image and the structure representation generated from the cartoon image since they should be the same. The system also perform a smoothing procedure by including a pre-trained VGG16Net to obtain another loss to remove high frequency noise in the output cartoon image from the generator.

In total, their cartoonizetion system achieved better mean quality score according to user study result, which is better than other cartoon generation GANs for now. Their system can well extract human face expression with cartoonized and smoothed structure, which we believe can help limit most of the noise that does not relate to facial expression.

## 3. Method

The original real human facial expression may be not so obvious to extract determing features for classification. We believe that by cartoonizing real human face images to more exaggerate cartoon faces, the key features will be intensified and more suitable for subsequent tasks. So we proposed a model combining cartoonization and VGG as in Figure 3.

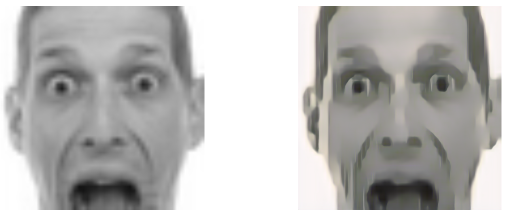

Figure 2. Real human face(left) and cartoonized face(right). (enlarged to show that the forehead wrinkles are removed)

### 3.1. Dataset Cartoonization and Augmentation

This work uses FER2013 dataset as the basic dataset for training and testing. We refer to the official training, validation, and test sets as introduced by the original paper [5]. FER2013 consists of 35888 images of 7 different emotions: anger, neutral, disgust, fear, happiness, sadness, and surprise.

The whole data preprocessing procedure is shown in Figure 4, including cartoonization, basic transformation and tencropping.

Wang et al.'s GAN system will be used to preprocess the images by cartoonizing real human facial expression image , such that to generate training, evaluation and test set of this work.

To boost the variability and balance the amount of images in different categories, we perform random basic image transformation to the cartoonized images in training. The cartoonized images are rescaled, horizontally and vertically shifted, and rotated to diversify training data. The rate of rescaling and shifting is random in the range $\pm$ 20 % of original scale and size, and rotating up to $\pm$ 10 degrees. Each of the augmentation is applied randomly and with a probability of 50 % . After the basic transformation, the image is then cropped into four corners and the central crop plus the horizontal flipped version of these. This procedure is called tencrop, and will increase the number of images by 10 times. After tencorpping, the images have a size of 40$\times$40, and random portions of each of the crops are erased with a probability of 50 %, and then normalized by applying a divition of 255 on each pixel.

### 3.2. VGG Model

VGGNet is a lightweight convolutional neural network architecture used to process large-scale image and perform pattern recognition and classification tasks with little dependency on hardware capability [7] [16]. We adopt a version of VGG net(Figure 5) to perform the classification task. The VGG network has 4 convolutional stages and 3 fully connected layers. The convolutional stages are capable of feature extraction, dimension reduction, and non-linearity [20]. One convolutional stage contains two convolutional blocks and a max-pooling layer. One convolution block is composed by a convolutional layer, an ReLU activation, and a batch normalization layer. Batch normalization here can accelerate learning process, reduce the internal covariance shift, and avoid gradient vanishing and gradient explosion [7] [6]. The fully connected layers are trained to classify the inputs as described by extracted features [1]. The ReLU activation between the first two fully connected layers and the third fully connected layer is a preparation for the third fully connected layer's classification.

$$l_n = -\sum_{c=1}^{C} y_{n,c} \log(softmax(x_{n,c})) \qquad (1)$$

For the loss function, we simply adopt the cross entropy loss to be our loss funtion(Eq.1) when comparing the predicted result and the correct label.

In Eq.1, $l_n \in \{l_1, l_2, ..., l_N\}$ is the cross entropy loss of a single predict result in a batch and $N$ is batch size, $C = 7$ is the number of classes, $x_{n,c}$ is the $nth$ predicted probrability of the class $c$, and $y_{n,c}$ is the correct label of $x_{n,c}$.

$$softmax(x_{n,c}) = \frac{\exp{(x_{n,c})}}{\sum_{i=1}^{C} \exp{(x_{n,i})}} \qquad (2)$$

After applying the softmax function(Eq.2), the loss function can be expressed as below:

$$l_n = -\sum_{c=1}^{C} y_{n,c} \log \frac{\exp{(x_{n,c})}}{\sum_{i=1}^{C} \exp{(x_{n,i})}} \qquad (3)$$

We adopt the mean of loss to represent the total loss in a batch.

$$l(x, y) = \frac{\sum_{n=1}^{N} l_n}{N} \qquad (4)$$

### 3.3. Fine Tuning

Our model will be trained with different hyperparameter settings, optimizers and learning rate schedulers to find the suitable ones that maximize model performance. We will firstly determine the optimal batch size with SGD as optimizer, 0.01 as learning rate, 0.5 as drop-out rate for the fully connected layers for 100 epochs. After the batch size has been determined, we then go back to select the best optimizer and learning rate scheduler on our model's performance. Finally, based on the trained model, we set up a final experiment to try to further improve model accuracy by adding validation data to training dataset or applying warm-restart to the learning rate.

## 4. Experiments

### 4.1. Dataset Preparation

We get Fer2013 dataset from Kaggle discussion held by the competition organizers places. We extract all the samples

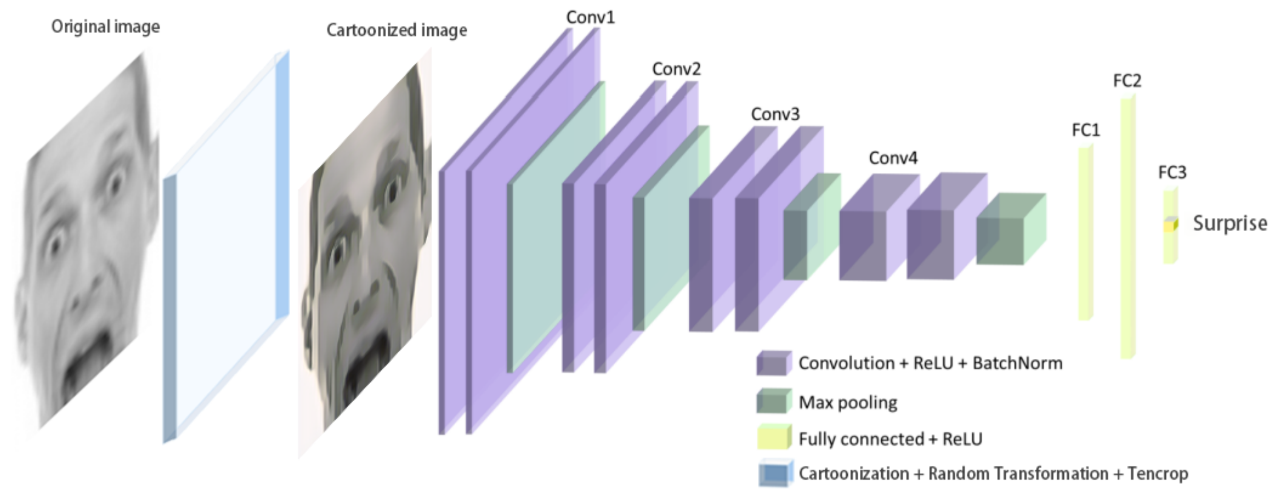

Figure 3. Backbone of our cartoonization+VGG model

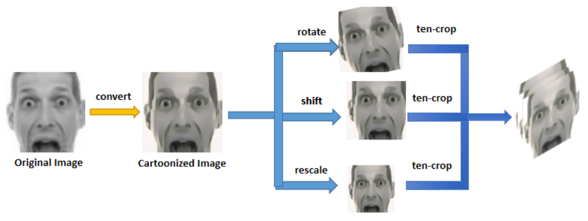

Figure 4. Data preprocessing procedure

and construct the input cartoonized image data and save them in a csv file. Before training, we load the csv file and peform data augmentation as described in previous section.

## 4.2. Model Training

Firstly, we have to determine the optimal optimizer and learning rate schedule. After the batch size has been optimized, we then go back to determine the best optimizer and learning rate scheduler on our model's performance. Finally, we then set up an experiment to try and fine-tune the trained model trained in previous step.

**Batch Size Selection.** We firstly run the model with SGD as optimizer, 0.01 as learning rate, 0.5 as drop-out rate for the fully connected layers for 100 epochs on batch size 16, 32, 64, 128, 192 and 256 respectively.

From this step, we can determine the optimal batch size for the following training and tuning procedure. We observe that 32 as batch size is the best choice as summarized in Tabel 1.

**Optimizer and LR Scheduler Selection.** The optimal optimizer and learning rate scheduler will be decided by running a grid search for 300 epochs since the LR Schedule may take effect on long run. All schedulers have an initial learning rate of 0.01, batch size 32 and dropout rate 0.5.

The grid search includes optimizers and learning rate schedulers, in which the former conatins SGD with Nesterov Momentum, Average SGD, Adam, Adadelta, and Adagrad, and the latter contains RLRP, CosineAnnealingLR, CosineAnnealingWarmRestarts, OneCycleLR and StepLR.

From this step, we can select some of the models that out performs others for the next tunning procedure. From Table 2, it is clear to see that some optimizers and learning rate schedulers have greater effect on this task over others. We can observe that the SGD+RLRP, ASGD+RLRP, Adam+RLRP and SGD+CosineWR have accuracy over 70%, so we pick these four combination for the next step.

## 4.3. Model Tuning

The model will be fine tuned by applying warm-restart to oscillate the learning rate and adding validation data to training dataset to further raise the test accuracy.

**Tuning With Oscillate Learning Rate**. We reload the parameters and train for another 50 epochs with 0.0001 as initial learning rate so that the update steps will not override the previous trained 300 epochs too much. We use Cosine Annealing and Cosine Annealing with Warm Restarts for both of them oscillate the learning rate back and forth to maximize the model accuracy in a small range. The second schedule may also take effect since the warm restarts would regularly reset the model's weights back to some good location during its updates.

By using this tuning procedure, we expect this step can bring model accuracy higher and further filter out the best

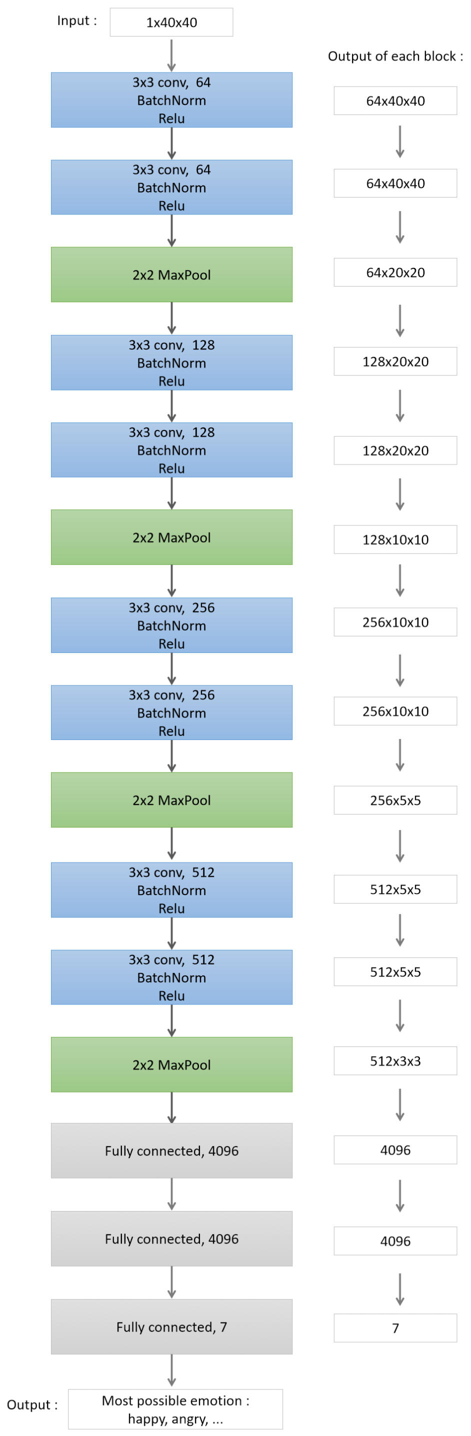

Figure 5. VGG part model structure

models. From Table 3, we can figure out the effectiveness of CosineWR or Cosine tuning. Compared with Table 2, some of the accuracy have raised to the extent of 0.8% such as

Table 1. Accuracy in 100 epochs with different batch size

| Batch size | Acc(%) |
|---|---|
| 16 | 69.9081 |
| 32 | **71.0783** |
| 64 | 70.8833 |
| 128 | 70.9390 |
| 192 | 70.1031 |
| 256 | 69.9638 |

SGD+StepLR, but others have little effect like ASGD+RLRP, whose best accuracy is the same in Table 2. We further select SGD+StepLR and SGD+RLRP as better models out of others.

**Tuning With More Data** We then ran a second variation of this experiment in which we combined the validation set into training. Larger dataset may let model have more data to learn from. We would like to get an evaluation of whether more data improves the model's performance.

The test set is unaltered and all other parameters and running environment are kept the same as the best model obtained in the CosineWR or Cosine tuning part, that is, run 300 epochs with more data, drop out rate 0.5 and initial learning rate 0.01, then another 50 training epochs with CosineWR or Cosine tuning with more data, drop out rate 0.5 and initial learning rate 0.0001.

From Table 4, we have observed an overall improvement on both models. By combining validation set into training set, SGD+StepLR has a 1.1% imporvement and SGD+RLRP has a 1.6% improvement. The model obtained in this tuning step has the highest total accuracy among all attempts in this work. We use SGD+RLRP and SGD+StepLR for confusion matrix and SGD+RLRP for saliency maps analysis.

## 4.4. Evaluation Method

**Confusion Matrix**. The confusion matrix can show the accuracy of each emotion and how the model mis-classify one image to another emotion. This matrix can help us find out which emotion can be classified well or bad.

**Saliency map**. We generate saliency maps to visualize the information captured inside the model in order to understand how it computes its prediction and differentiates between and classifies different facial image emotions on cartoonized images. To be more precise, we randomly select 7 images from each class to enerate saliency maps to visualize the information captured inside the model. We also refer to Yousif and Chen's VGGNet to generate a control group using real human images [7].

Within the saliency maps, we can compare the difference between the way model processes the real human images and the cartoonized images.

Table 2. Accuracy in 300 epochs with different optimizers and LRschedulers

| Acc(%) LRscheduler / Optimizer | RLRP | StepLR | OneCycleLR | CosineWR | Cosine |
|---|---|---|---|---|---|
| SGD | **70.4932** | **70.8833** | 69.2115 | **70.1867** | 69.2951 |
| ASGD | **70.1588** | 68.9886 | 66.7317 | 61.6606 | 68.9050 |
| Adam | **70.3817** | 67.5397 | 69.9638 | 68.4035 | 68.8493 |
| Adagrad | 64.9485 | 64.1126 | 66.3137 | 66.9824 | 60.1839 |
| Adadelta | 68.7657 | 68.2920 | 67.3168 | 68.4035 | 65.0599 |

Table 3. Accuracy of CosineWR or Cosine tuning with initial learning rate 0.0001

| Combination | Accuracy of CosineWR(%) | Accuracy of Cosine(%) |
|---|---|---|
| SGD+StepLR | **70.8833** | **71.6077** |
| Adam+RLRP | 70.3817 | 70.4653 |
| ASGD+RLRP | 70.1588 | 70.4374 |
| SGD+RLRP | **70.7718** | **70.9947** |
| SGD+CosineWR | 70.2424 | 70.1867 |

## 5. Results

**Confusion Matrix**. The matrix shown in Figure 6 and 7 shows the confusion of different emotion for training data of SGD+RLRP and SGD+StepLR. We can observe that in both matrics, disgust, happy and suprise have very high accuracy while anger, fear, sad and neutral, in some cases, are confused with each other.

**Saliency Map**. We randomly select 7 images from each class to enerate saliency maps to visualize the information captured inside the model. We also refer to Yousif and Chen's VGGNet to generate a control group using real human images [7]. Figure 8 shows the saliency map result. From the comparasion, we can observe that the weight parameters in two models have some difference in distribution on the image. Yousif and Chen's VGGNet has weight over all pixels on real face and value more on wider skin parts like cheek and forhead, while our model focus more on pixels that represent facial features like mouth and nose. This can be clearly seen in (b) (c) and (e), where more features are on nose and mouth and less on face. But we also observed that the features in (f) and (g) also widely spread on cheek and forehead just like the result in Yousif and Chen's VGGNet [7]. The observation can be explained by the cartoonization, which will remove the skin details like wrinkles and freckles, thus lowering the weight in pixels that belongs to those areas. This change of feature fits our precision that obtains best accuracy on surprise, happy and disgust, which exactly corresponding to (b), (c) and (e). This result also prove our initial consideration that skin details like wrinkles and freckles are unimportant when performing facial image emotion recognization.

However, the cartoonization for infant faces will not create too much difference for the infant faces are usually very smooth. In that case, the weight distribution are similar in both model as seen in (f) and (g). To avoid this situation, other cartoonization method should also be tested to see if they can have better performance in the future. There are also some obvious mistakes in the saliency maps, most of which are due to incorrectly regarding some background pixels as features. Maybe we need a method to generate mask on each image to mask out the background. This should help model more effectively identify facial features and avoid all useless information.

### 5.1. Performance Comparison

Table 5 summarizes benchmark on Fer2013 testset. Our method out performs most methods and ranks third after [7] and [10] with slight gap on total accuracy.

Table 6 shows the detailed accuracy rate of the top 3 models. Our model has similar accuracy on happy and disgust, and best accuracy on surprise. This shows that training with cartoonized facical images has better classification capability on recognizing certain types of emotions.

Table 4. Accuracy of CosineWR or Cosine tuning with more data

| Combination | Accuracy of CosineWR(%) | Accuracy of Cosine(%) |
|---|---|---|
| SGD+StepLR | 72.0256 | 71.8863 |
| SGD+RLRP | **72.3321** | 72.2207 |

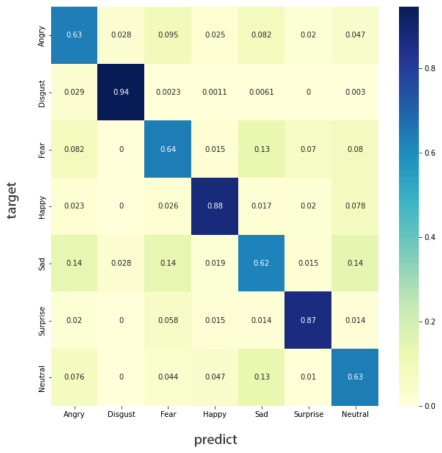

Figure 6. Confusion matrix of SGD+RLRP

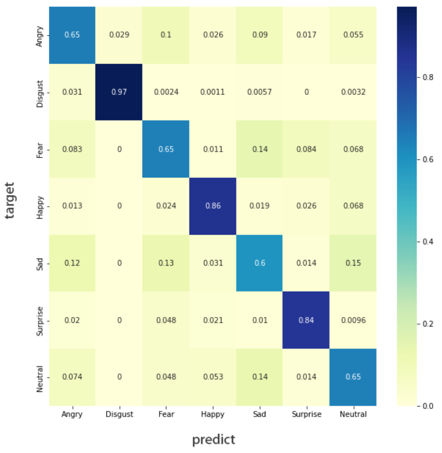

Figure 7. Confusion matrix of SGD+StepLR

Table 5. Fer2013 public testset benchmark

| Method(single model) | Acc(%) |
|---|---|
| Attentional ConvNet [11] | 70.02 |
| Inception [13] | 71.6 |
| Xception+Ad-Corre Loss [3] | 72.03 |
| CNN [18] | 72.16 |
| VGG+Cartoonize(this work) | **72.33** |
| VGGNet [7] | 73.28 |
| ResNet18 [10] | 73.70 |

## 6. Conclusion

This paper proposes an approach to use cartoonized image to train an variant of VGGNet for facial image emotion recognization. We cartoonize all images in Fer2013 dataset using a GAN system [19], then thoroughly tune all hyperparameters including batch size, optimizer and learning rate scheduler. After that, we try to fine-tune the model with oscillate learning rate and add validation set to training set. The best testing classification accuracy we achieved is 72.33 %, which is top 3 total classification accuracy on FER2013 dataset in single network. We also compared the accuracy on each class that our model has obtained with other outstanding models, and achieve best accuracy on surprise and disguist. We analyze the features our model extracts using saliency maps, which further prove that our model can effectively ignore unimportant facial features such as wrinkles and freckles on face, thus proving the effect of cartoonization on emotion classification tasks.

For future work, we plan to explore different cartoonizing techniques and combine them with different deep learning architectures and larger emotion-releated datasets to further explore the effect cartoonization on facial image emotion recognition tasks.

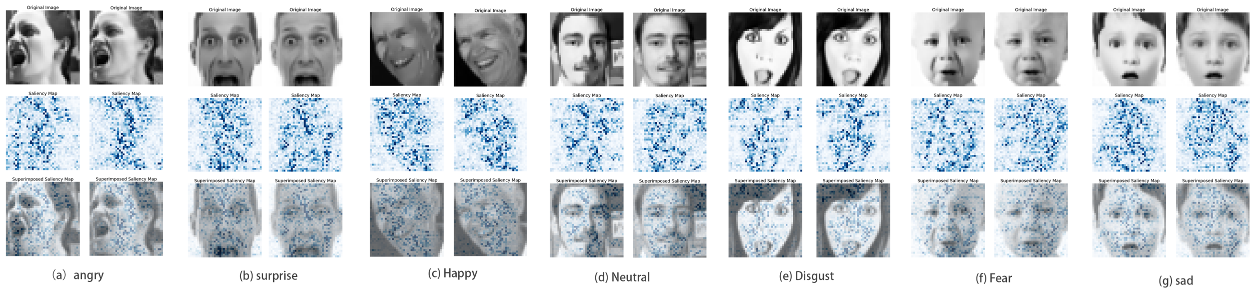

Figure 8. Saliency map of all emotions on cartoonized images(left) and real images(right). More features are on nose and mouth and less on cheek and forehead In (b) (c) (e). But in (f) and (g) features on cheek and forehead do not decrease comparatively, since facial images of children and infants have less change after cartoonization than others.

Table 6. Accuracy in each emotion class

| | VGGNet [7] | ResNet18 [10] | VGG+Cartoonize(this work) |
|---|---|---|---|
| Anger | 0.66 | 0.65 | 0.63 |
| Disgust | 0.64 | 0.94 | **0.94** |
| Fear | 0.57 | 0.68 | 0.64 |
| Happy | 0.89 | 0.88 | 0.88 |
| Sad | 0.65 | 0.64 | 0.62 |
| Surprise | 0.85 | 0.82 | **0.87** |
| Neutral | 0.71 | 0.65 | 0.63 |

## References

[1] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 international conference on engineering and technology (ICET)*. Ieee, 2017, pp. 1–6. 3

[2] W.-S. Chin, Y. Zhuang, Y.-C. Juan, and C.-J. Lin, "A learning-rate schedule for stochastic gradient methods to matrix factorization," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2015, pp. 442–455. 2

[3] A. P. Fard and M. H. Mahoor, "Ad-corre: Adaptive correlation-based loss for facial expression recognition in the wild," *IEEE Access*, vol. 10, pp. 26 756–26 768, 2022. 7

[4] M.-I. Georgescu, R. T. Ionescu, and M. Popescu, "Local learning with deep and handcrafted features for facial expression recognition," *IEEE Access*, vol. 7, pp. 64 827–64 836, 2019. 1

[5] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee *et al.*, "Challenges in representation learning: A report on three machine learning contests," in *International conference on neural information processing*. Springer, 2013, pp. 117–124. 1, 3

[6] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456. 3

[7] Y. Khaireddin and Z. Chen, "Facial emotion recognition: State of the art performance on fer2013," *arXiv preprint arXiv:2105.03588*, 2021. 1, 2, 3, 5, 6, 7, 8

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012. 2

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. 1

[10] LetheSec, "Fer2013-facial-emotion-recognition-pytorch," https://github.com/LetheSec/Fer2013-Facial-Emotion-Recognition-Pytorch, 2021. 6, 7, 8

[11] S. Minaee, M. Minaei, and A. Abdolrashidi, "Deep-emotion: Facial expression recognition using atten-

tional convolutional network," *Sensors*, vol. 21, no. 9, p. 3046, 2021. 7

[12] R. Pecoraro, V. Basile, V. Bono, and S. Gallo, "Local multi-head channel self-attention for facial expression recognition," *arXiv preprint arXiv:2111.07224*, 2021. 1, 2

[13] C. Pramerdorfer and M. Kampel, "Facial expression recognition using convolutional neural networks: state of the art," *arXiv preprint arXiv:1612.02903*, 2016. 7

[14] E. Pranav, S. Kamal, C. S. Chandran, and M. H. Supriya, "Facial emotion recognition using deep convolutional neural network," *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pp. 317–320, 2020. 1, 2

[15] D. Shah, K. Chavan, S. Shah, and P. Kanani, "Real-time facial emotion recognition," 10 2021, pp. 1–4. 1

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 3

[17] R.-Y. Sun, "Optimization for deep learning: An overview," *Journal of the Operations Research Society of China*, vol. 8, no. 2, pp. 249–294, 2020. 2

[18] A. Vulpe-Grigoraşi and O. Grigore, "Convolutional neural network hyperparameters optimization for facial emotion recognition," in *2021 12th International Symposium on Advanced Topics in Electrical Engineering (ATEE)*. IEEE, 2021, pp. 1–5. 7

[19] X. Wang and J. Yu, "Learning to cartoonize using white-box cartoon representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 7

[20] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833. 3