Using Transformer and Involution for facial expression recognition

Zicheng Liu

Research School of Computer Science, Australian National University
u6924878@anu.edu.au

Abstract. Facial expression plays an important role in our daily life. Automatic facial expression recognition has attracted increasing attention in computer vision field. However, the recognition result is affected significantly by pose variations and occlusion, especially in real world scenarios. To address this challenge, we conduct experiments using recently proposed Involution kernel and Vision Transformer on SFEW dataset. Experiments demonstrate that by combining Involution kernel and Vision Transformer, our model achieves accuracy over 0.6.

Keywords: Facial expression recognition · Vision transformer · Involution.

1 Introduction

Facial expression is crucial in our daily communication. Nowadays, facial expression has potention application in many fields, such as psychology, artificial intelligence etc. Facial expression recognition has drawn increasing attention in computer vision field. However, when it comes to real world scenarios, recognizing facial expression becomes a challenging task since the result is significantly affected by optical condition, pose and perspective. Moreover, many facial expression datasets are collected in lab-controlled environments, which can not be used in real world.
To address this challenge, many researchers have done experiments on this task. Pourmirzaei et al.[7] implemented a method which uses self-supervised co-training to improve performance on facial representation. Based on multi-task learning of lighweight neural networks, Savchenko[9] presented several models on facial expression and attributes recognition. Wang et al.[11] proposed a region attention network for robust facial expression recognition. These methods have achieved great performance on facial expression recognition task. Recently, vision transformer[3] and involution[6] has attracted great attention in computer vision community. In this paper, we combine these two methods by using a parallel architecture. We conduct experiments on SFEW[2] dataset. Experiments shows that our model holds a good performance on facial expression recognition task.

2 Related Work

2.1 Involution

Convolution neural network has been the mainstream of deep neural network for computer vision for years. Convolution kernel has two key properties, spatial-agnostic and channel-specific. Thanks to these two properties, convolution kernel holds translation invariance and high efficiency. On the contrary, spatial-agnostic property restricts its ability to adapt to diverse visual patterns on different locations. Besides, the small receptive field constrains the ability to capture spatial relation in a single image. Moreover, researchers have proved that there exists significant inter-channel redundancy in convolution kernel.
To conquer these limitations, Li et al. proposed a novel operation called involution, whose properties are symmetrically inverse inherent compared to convolution, which are spatial-specific and channel-agnostic. To be more specific, involution kernel is distinct along spatial dimensions while shared across channels. An involution kernel is generated solely dependent on a specific spatial location, and the redundancy of kernels are shared along the channel dimensions.

2.2 Vision Transformer

Transformer was first proposed by Vaswani et al.[10] in 2017 for natural language processing tasks. Once after proposed, it has drawn great attention in the machine learning community, including computer vision community. Recently, Dosovitskiy et al.[3] proposed Vision Transformers which applies transformer architecture on images. The pure self-attention vision transformer architecture achieved great success on ImageNet classification challenge[8], outperforming other state-of-the-art models.

3 Method

3.1 Network architecture

We use involution network and vision transformer in two parallel branches. Then output feature maps from these two branches are fused using 1x1 convolution. We then do the final prediction using this fused feature map. We introduce the detailed topology of these two branches below, and the detailed network topology is shown in Figure 1.

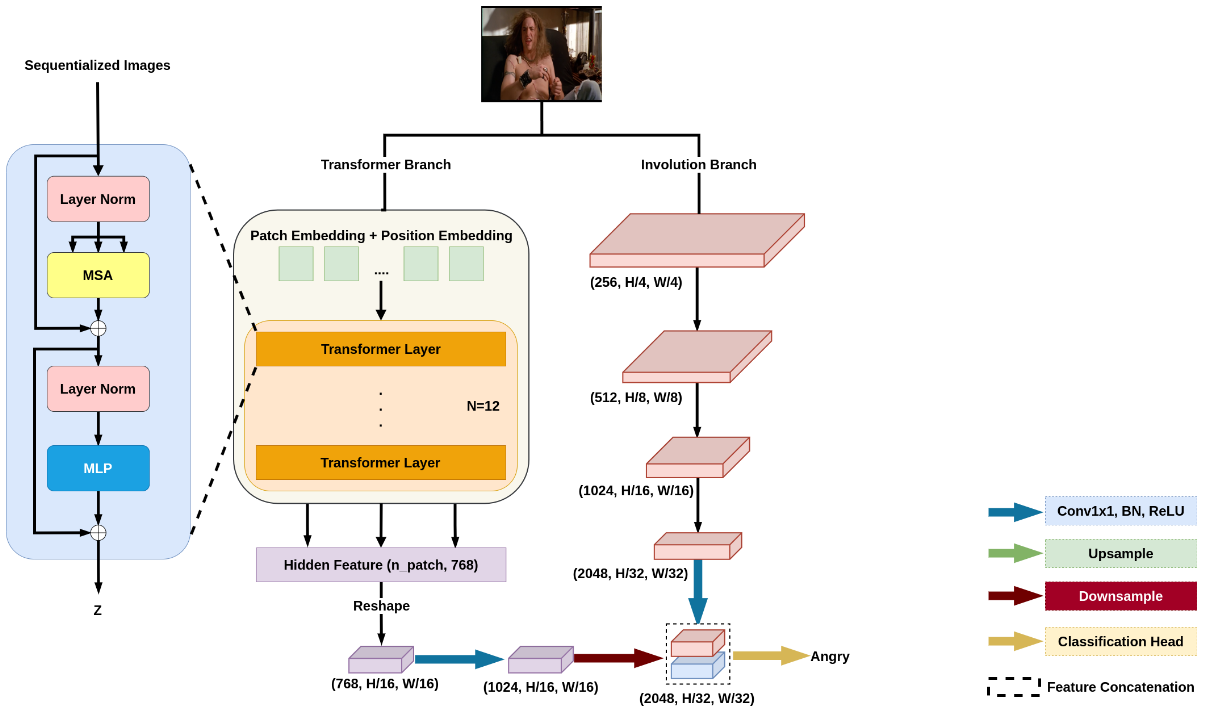

Fig. 1. Overview of network architecture

Involution network In the original paper, Li et al.[3] design a simple scheme for involution kernel. Involution kernel $H \in R^{H\times W\times K\times K\times G}$ are designd to apply a transformation with inverse characteristics compared with convolution kernels. To gain a deep insight of involution, an involution kernel $H_{i,j,\cdot,\cdot,g}, g = 1, 2, \cdot, \cdot, G$ is designed specifically for pixel $X_{i,j}$ but shared over channels where G is the number of groups, and each group shares the same involution kernel. Like convolution kernel, the output of involution kernel is also derived by a multiply operation and add operation, which can be defined as

$$Out_{i,j,k} = \Sigma_{(u,v)\in\Delta_K} H_{i,j,u+[K/2],v+[K/2].[KG/C]} X_{i+u,j+v,k} \tag{1}$$

As we have shown before, the shape of involution kerels H depend on the shape of the input feature map X. Thus, we could generate involution kernels by using the input feature map (or using part of it). As a result, output kernels can aligns to the input feature map (on spatial dimensions). The kernel generation function can be described as

$$H_{i,j} = \Phi(X_{\psi_{i,j}}) \tag{2}$$

where $\psi_{i,j}$ is the set of pixels which involution kernel $H_{i,j}$ is conditioned on.
To keep things easier, authors choose to generate $H_{i,j}$ from a single pixel $X_{i,j}$, and the generation function takes the following form

$$H_{i,j} = \Phi(X_{i,j}) = W_1\sigma(W_0 X_{i,j}) \tag{3}$$

where $W_0 \in R^{\frac{C}{r}\times C}$ and $W_1 \in R^{(K\times K\times G)\times \frac{C}{r}}$ are two linear transformations which construct a bottleneck structure here.
Now, involution kernels for a feature map can be generated using the generation function. In a nutshell, involution kernels are able to capture the relation ship in wider spatial range, and weights for different locations can be adaptively allocated. As a result, involution ahcieves better performance while keeping lower computational cost compared with convolution.

In our paper, we follows the procedure descrided in [involution]. To be more specific, Li et al. proposed a novel architecture namely RedNet which mirrored the design of the famous ResNet[4]. Authors replaced 3x3 convolution at all bottleneck positions in the stem. Specifically, they replaced 3x3 convolution kernels with 3x3 or 7x7 involution kernels. The other convolution kernels used for projection and fusion (mostly 1x1) are remained. We use a 50 layer RedNet for our task, namely RedNet50. Pre-trained weights on ImageNet[8] dataset is used so that we can better fine-tuning our model.

Vision Transformer To apply transformers to images, we need to transfer images into 1D sequences. For an image $x \in R^{H \times W \times C}$, we split into N patches where $N = \frac{HW}{P^2}$, and each patch has a size $P \times P$. We then reshape the patchlized image into $x_p \in R^{N \times (P^2 \cdot C)}$. Therefore, we now have this sequence as our input.
After we do the patchlization, a linear projection is applied to project all patches into D dimensions. Besides, position embeddings are added to preseve the positional information. This procedure can be derived as

$$z_0 = [x_p^1 E, x_p^2 E, \cdot, \cdot, \cdot, x_p^N E] + E_{pos} \tag{4}$$

where $E \in R^{(P^2 \cdot C) \times D}$, $E_{pos} \in R^{N \times D}$. Then L layers of multihead self-attention (MSA) and multilayer perceptron (MLP) are applied to encoded embeddings. Moreover, LayerNorm (LN) is applied before every block. Residual connections are applied after every block[12]. MLP layer use a GELU[5] activation function. This procedure can de described as

$$z_l^{'} = MSA(LN(z_{l-1})) + z_{l-1} \tag{5}$$

$$z_l = MLP(LN(z_l^{'})) + z_l^{'} \tag{6}$$

where $l = 1 \cdots L$. Now we can get the encoded features through vision transformer.
In our expriments, hybrid architecture is used before vision transformer. We use ViT-B_16[3] configuration. ViT-B_16 contains 12 MSA and MLP layers, each MSA layer contains 12 heads. The output hidden size is 768. Also, pre-trained weights on ImageNet[8] dataset are used for better fine-tuning.

3.2 Dataset

In our paper, we use a dataset named Static Facial Expression in the Wild (SFEW). The dataset is originally proposed by Dhall et. al.[2] SFEW dataset is developed by selecting frames from another dataset named Acted Facial Expressions in the Wild (AFEW)[1], which is extracted from movies.
In the past, face emotion data was mostly lab-controlled data, the method developed on such data is not that robust and can not be applied to real world application. Unlike those lab-controlled datasets, SFEW dataset covers face emotion images from varied head poses, different ages, occlusions and varied focus, which makes the dataset close to real world. It contains 700 images and have been labelled for 7 classes, which are angry, disgust, fear, happy, sad, surprise and neutral. However, we only have 675 images in our experments. We randomly split the train dataset and test dataset. Sepecifically, 90% of SFEW dataset are used for training (607 images), and the rest 10% are used for testing (68 images).

3.3 Implementation Details

All images are reszied to 352x352. Image augmentation techniques including affine transforms, rgb shift, random brightness and contrast are applied on the training data.
To train our model, we use the followng configuration.

1. We set the batch size to 8 with accumulation steps 2, which equals to batch size 16.
2. The model is trained for 50 epochs with a start learning rate at 5e-4. Polynomial learning rate scheduler is used to reduce the learning rate during training.
3. We use AdamW optimizer with a weight decay at 5e-4.

We implement our model using Pytorch 1.7.1. All models are trained and inferenced using a Rtx 2080Ti GPU. The system we use is Ubuntu 20.04.

4 Experimental Results

To provide a deeper insight into the model performance, we employ several metrics for quantitative evaluation, including Precision, Recall, F1-Score and Accuracy. They should be caculated as follows:

$$\begin{aligned} Accuracy &= \frac{tp + tn}{tp + fp + fn + tn} \\ Precision &= \frac{tp}{tp + fp} \\ Recall &= \frac{tp}{tp + fn} \\ F1 - Score &= \frac{2 \times (Recall \times Precision)}{Recall + Precision} \end{aligned}$$

Here tp =True positive, fp =False positive, tn =True negative, fn =False negative. Also, according to the SPI baseline, the base classification accuracy is 0.19. We compare our methods with the baseline and other state-of-art methods. The detailed quantitavie results of our model are shown in Table 1.

Quantitative results illustrate that our model holds a good performance on SFEW dataset. Compared with other methods, our model achieves accuracy over 0.64, which outperforms other SOTA models by a large margin (over 0.8), which demonstrates that our models holds a strong learning ability on facial expression recognition task.

Table 1. Quantitative results on SFEW dataset

Method	Accuracy	Precision	Recall	F1-Score
Baseline	0.19	-	-	-
Island Loss	0.5252	-	-	-
RAN	0.564	-	-	-
Involution+ViT	0.6471	0.6847	0.6565	0.6615

5 Conclusion

Inspired by involution and vision transformer, we combine these two techniques together using a parallel architecture. To prove the learning ability of our model, we conduct experiments on facial expression recognition task. Experiments results once again demonstrate the strong representation ability of involution kernel and vision transformer, which can significantly improve the model's performance compared with traditional CNN-based model. We hope our experiments may contribute to the community and push the boundary of facial expression recognition.

References

1. Dhall, A., Goecke, R., Lucey, S., Gedeon, T.: Acted facial expressions in the wild database. Australian National University, Canberra, Australia, Technical Report TR-CS-11 **2**, 1 (2011)
2. Dhall, A., Goecke, R., Lucey, S., Gedeon, T.: Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). pp. 2106–2112. IEEE (2011)
3. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale (2020)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
5. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016)
6. Li, D., Hu, J., Wang, C., Li, X., She, Q., Zhu, L., Zhang, T., Chen, Q.: Involution: Inverting the inherence of convolution for visual recognition (2021)
7. Pourmirzaei, M., Esmaili, F., Montazer, G.A.: Using self-supervised co-training to improve facial representation. arXiv preprint arXiv:2105.06421 (2021)
8. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
9. Savchenko, A.V.: Facial expression and attributes recognition based on multi-task learning of lightweight neural networks (2021)
10. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need (2017)

11. Wang, K., Peng, X., Yang, J., Meng, D., Qiao, Y.: Region attention networks for pose and occlusion robust facial expression recognition. IEEE Transactions on Image Processing **29**, 4057–4069 (2020)
12. Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D.F., Chao, L.S.: Learning deep transformer models for machine translation. arXiv preprint arXiv:1906.01787 (2019)