Custom facial keypoint detection

Omid Jadidi
University Of Trento
omid.jadidi@studenti.unitn.it, rkomid5@gmail.com

1. Introduction

Facial key points prediction is the task where a neural network predicts the key points in a face image. Facial key points are relevant for a variety of tasks, such as face filters [11], emotion recognition [5], and so on. As far as I know most of the pre-trained models can predict only a certain set of key points consist of nose, eyes, lips,eye brows and lower skin, where the annotations are available [12]. In this paper we are looking to conduct a way which can predict key points not only for certain face attributes but for all attributes in the absence of annotations.

1.1. Problem Statement

The task is predicting key points for each attributes of face by showing the image of the face only. First problem is we don't have labeled key points for all attributes of face and second problem is related to some special attributes for instance hair, key points only can not be a good representative for the hair and can cause issues like predicting all hair key points on center of image and so on. So the best label for this task is attributes masks as input. I am using ClebAMask-HQ [3] as my input for prediction 448*448 input images and 19 classes of attributes mask.

1.2. Previous Methods

There are two main lines for predicting key points. First by using fixed number of key points as labels and Network [9] will predict the points by minimizing RMSE of the predicted key points. model architecture can be inherited from famous networks like LeNet, LeNetWithDropout, and SimpleVGGNet. We can not expect good result from this method, key points are not well predicted and still just few attributes are considered.Second approach is using CNN model but instead of key points as label, we convert key points to heatmaps and use those as label. This method will allow us to predict many attributes with good accuracy in predicting. For instance FAN network [12] can reach very high performance in predicting some facial attributes. but still there isnt any trained network for all face attributes.

1.3. Proposed method

My proposed method is based on CelebAMask-HQ [3] face parsing method but instead of using BiSeNet [13] I used Residual blocks and ResNet [2] architecture with some modification for less model complexity and computational power. This network can predict all face attributes consist of 19 class as mask that covers them and finding edge of them using canny edge method. This method will allow us to predict all important key points not just few of them.

1.4. Dataset

CelebAMask-HQ [3] is a large-scale face image dataset that has 30,000 high-resolution face images selected from the CelebA dataset by following CelebA-HQ. Each image has segmentation mask of facial attributes corresponding to CelebA. The masks of CelebAMask-HQ were manually-annotated with the size of 512 x 512 and 19 classes including all facial components and accessories such as skin, nose and so on.

2. Technical Approach

2.1. Data Augmentation

using a variety of data augmentation techniques, helps generalize our model. The most effective ones for training purpose were Image normalizing, Color Jitter, Horizontal flip, random scale, random crop. First normalizing 3 channel of image colors to the fixed mean and std of (0.485, 0.456, 0.406), (0.229, 0.224, 0.225). Color Jitter will randomly change the contrast, saturation and brightness of image. Horizontal flip will mirror the image. Random scale will randomly scale the image and label. and RandomCrop will crop randomly the part of the image.Output shape is (448,448) and each time picture is unique in the training mode.

2.2. Architectures

As base architectures for the proposed network we consider ResNet18. For easier explanation we seperate the network with 3 main block. Basic block consist of residual

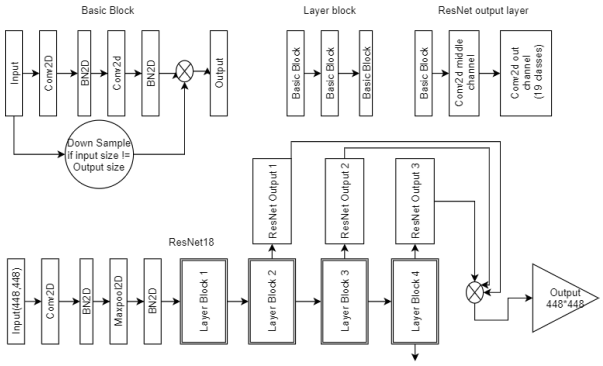

Figure 1. Detailed Architectures of our network with base of ResNet18

block (conv2d(64 channel),BN,conv2d(64 in,128 out),BN, stride 2 and padding 3), Second is Layer block consist of 3 basic blocks and third is ResNet output layer consist of basic block, conv2d in middle and conv2d with 19 channel as output (same number of our classes 19). ResNet18 will start with Conv2d (3in, 64 out, kernel size 7 ,stride = 2 , padding 3), BN,Max pooling and 4 Layer block, block1(64 in 64 out,stride 1 padding 1), block2 (64 in,128 out,stride 2 padding 3),block3 (128 in,256 out,stride 2 padding 1),block4 (256 in,512 out,stride 2 padding 1). The output of the network is 3 feature which has been extracted from layers 2,3,4. each feature has 19 channel equal to number of classes. Each channel of resnet output layer will focus on specific class. If value which assigned to a pixel is higher, it mean its more probable. Summing these three resnet output layer will gives us good view over our classes. To achieving the final masks in one 448*448 picture we simply return the argmax of classes in final summed feature for all pixels.

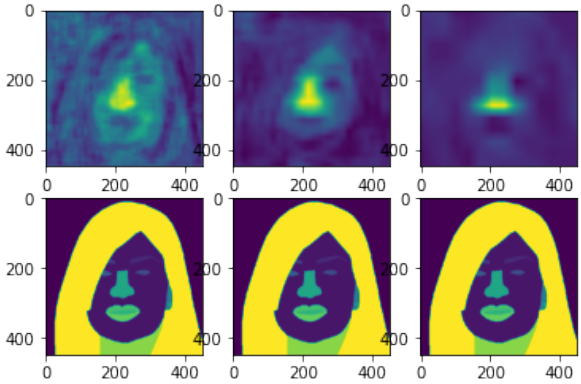

Figure 2. Picture depict 3 feature extracted from different layer of ResNet predicted the position of nose From left to right Output of Resnet output layers 2, 3 and 4

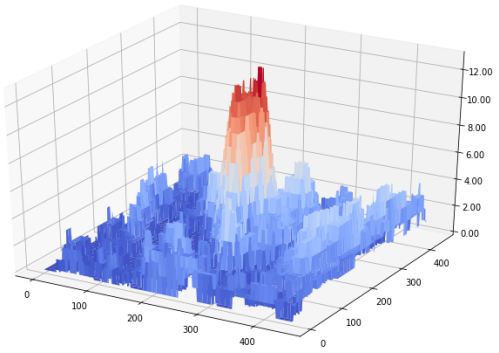

Figure 3. 3D representation of predicted class (nose) by summing values of 3 features.Higher value means higher probability of class

2.3. Loss function

For loss function I used Ohem (Online Hard Example Mining) cross entropy implementation [6]. It works nicely in the Stochastic Gradient Descent (SGD) paradigm, simplifies training by removing some heuristics and hyper parameters, also leads to better convergence (lower training set loss). The main function is setting a threshold(in my case 0.7), calculate the cross entropy for batch of size 16*488*488, sort the loss by its value and check last value of 448*448 items in a sorted loss array, if its above threshold return average of all values above threshold if not return only average of first 448*448 values. This method will allow network to not only focus on miss predicted pixels with large loss values but also consider classes with smaller loss value. after many iteration this will lead to better convergence.

2.4. Optimizer

Stochastic Gradient Decent will do magic. For fine tuning the optimizer I choose the parameters based on the works of Bisenet [13]. momentum 0.9, weight decay 5e-4, Learning rate start 1e-2. Gradient descent with weight decay is defined by the following update rule:

$$Z^{t+1}=(1-\eta*\beta)Z_t-\eta*\nabla L(Z_t) \quad (1)$$

, where Beta defines the rate of the weight decay per step and eta is the learning rate. In this case, weight decay is equivalent to L2 regularization. For the learning rate I used poly strategy with 100 warm up step and power of 0.9.The basic form of the polynomial learning rate policy is

$$Lr = Lr_0 * (1 - \frac{i}{T_i})^{power}. \quad (2)$$

Here, Lr0 is the initial or base learning rate, i denotes number of iterations and Ti is the total number of iterations. In

case of polynomial learning rate policy, Ti is equal to total number of epochs times number of iterations in a epoch. Power term controls the shape of learning rate decay [4].

2.5. Mask edge

By predicting the masks(classes) for each pixel we need to convert them to key points by finding edge of each mask. This can be done by Finding Intensity Gradient of the Image. Image is then filtered with a Sobel kernel [10] in both horizontal and vertical direction to get first derivative in horizontal direction (Gx) and vertical direction (Gy). From these two images, we can find edge gradient and direction for each pixel as follows:

$$EdgeGradient(G) = (G_x^2 + G_y^2)^{-}\frac{1}{2}$$

$$Angle(\theta) = (tan^{-}1)(\frac{G_y}{G_x})$$

Gradient direction is always perpendicular to edges. It is rounded to one of four angles representing vertical, horizontal and two diagonal directions. After getting gradient magnitude and direction, a full scan of image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, pixel is checked if it is a local maximum in its neighborhood in the direction of gradient. Then, we need two threshold values, minVal and maxVal. Any edges with intensity gradient more than maxVal are sure to be edges and those below minVal are sure to be non-edges, so discarded. Those who lie between these two thresholds are classified edges or non-edges based on their connectivity. If they are connected to "sure-edge" pixels, they are considered to be part of edges. Otherwise, they are also discarded. The whole process is called canny edge detection [8].

2.6. Result

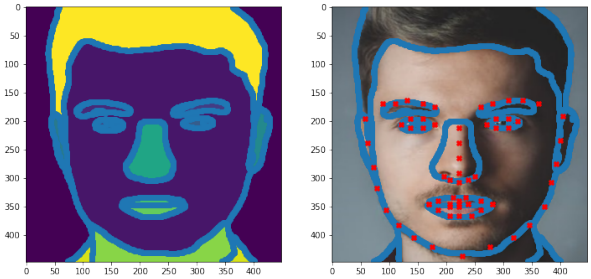

Figure 4. Left: Parsed masks with proposed network — Right: predicted points(Blue) and also Red points are predicted with FAN network [1]

Final results have been achieved after training 4000 iteration with 16 batch each time. There is no baseline for comparison because no one have predicted all attributes in one model yet, but still we can compare it with few well known face parser models like ResNet with attention architecture [7] with 4000 iteration too. There are one more model like Bisenet which can reach state of art performance but because of lack of computational power(need to be train more than 80000 it) I exclude it from comparison. For comparison I'm using same setting as mentioned above and take loss value as metric and for validation I'm using a set consist of 16 image which haven't been shown to the model from CelebAMask-HD data set. The table of result will show our result is near the same as of the best known networks but with much faster speed of training and less complexity.

Table 1. Final Loss Result

Table			
Model	Iteration	Training Loss	Validation loss
ResNet18	4000	2.87	2.73
ResNet with attention	4000	3.19	3.24

References

[1] A. Bulat and G. Tzimiropoulos. Super-fan: Integrated facial landmark localization and super-resolution of real-world low resolution faces in arbitrary poses with gans, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[3] C.-H. Lee, Z. Liu, L. Wu, and P. Luo. Maskgan: Towards diverse and interactive facial image manipulation, 2020.

[4] P. Mishra and K. Sarawadekar. Polynomial learning rate policy with warm restart for deep neural network. In *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, 2019.

[5] S. Shojaeilangari, W. Yau, K. Nandakumar, J. Li, and E. K. Teoh. Robust representation and recognition of facial emotions using extreme sparse learning, 2015.

[6] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining, 2016.

[7] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification, 2017.

[8] Wikipedia contributors. Canny edge detector — Wikipedia, the free encyclopedia, 2020.

[9] Wikipedia contributors. Lenet — Wikipedia, the free encyclopedia, 2020.

[10] Wikipedia contributors. Sobel operator — Wikipedia, the free encyclopedia, 2020.

[11] H. Yagou, Y. Ohtake, and A. Belyaev. Mesh smoothing via mean and median filtering applied to face normals.

[12] J. Yang, A. Bulat, and G. Tzimiropoulos. Fan-face: a simple orthogonal improvement to deep face recognition, 2020.

[13] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation, 2018.