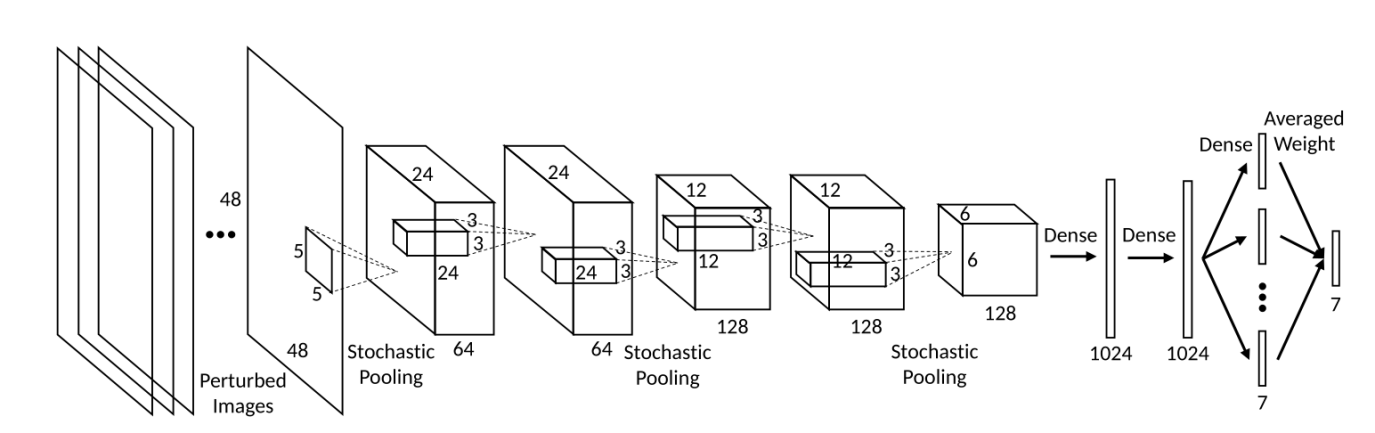
Method we will use:

Face Processing:

Face Processing is a crucial step for better recognition performance. It can remove irrelevant noise and integrate all faces to the same domain. In our project, we plan to resize all detected faces to 48 X 48 and transform them to grayscale. Then, we will use standard histogram equalization to process these face images, then removing unbalanced illumination by a linear plane fitting. At last, the image pixel values are normalized to a zero mean and unit variance vector.

Classification Module:

The classification Module we plan to use is the ensemble of multiple deep convolutional neural networks (CNN). Each CNN model is initialized randomly and pre-trained on the dataset. In order to improve classification performance, we introduce random perturbations and voting in our convolutional neural networks (CNN) architecture. The diagram of our CNN architecture lies below:



You can see that this neural network contains five convolutional layers, three stochastic pooling layers and three fully connected layers. The input to the network is the preprocessed 48 × 48 faces. Convolutional layers are used to feature extraction. And in our project we use stochastic pooling instead of max pooling. Stochastic pooling is randomly sampling a response based on the probability distribution obtained by normalizing the responses, which means the probability of being selected with a large element value is also large. The nonlinear mapping functions for all convolutional layers and fully connected layers are set as rectified linear unit (ReLU):

By the way, the fully connected layers contains dropout, which is another method of randomization.

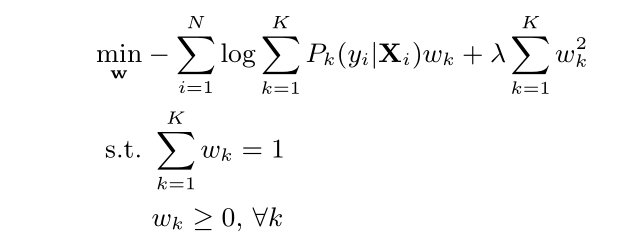
Each CNN model will be pre-trained on the dataset. We will try to set an initial network learning rate and a minimum learning rate. Samples from each training epoch are randomly selected from the training set and with random perturbation. The learning rate can be adjusted actively through the learning period. (Specific number to adjust learning rate can be discussed again) After all epochs are finished, we select the neural network from the epoch with the best training accuracy as our final pre-trained model.

What’s more, we need to fine-tune our network on the training set. In order to avoid overfitting, we try to freeze the parameters of all the convolutional layers and only allow the update of parameters at the fully connected layers. We know that a slightly larger learning rate helps to reduce the risk of trapping at local minima and benefits the fine-tuning performance. So in our project we can slightly increase the number of learning rate.

In order to learn the ensemble weights w, we train multiple convolutional neural networks and output their training responses. There are two optimization frameworks to consider in our project:

1. Optimal Ensembled Log Likelihood Loss

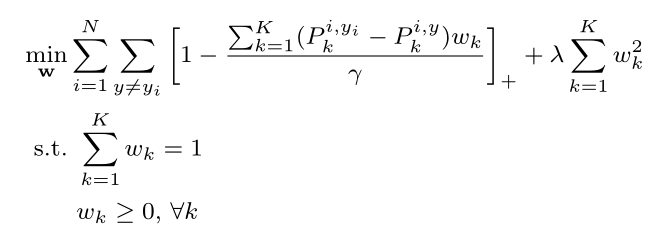
Here is the objective function:



N is the number of training samples, and K is the number of networks. Pk (y|Xi ) is the kth network output response on the yth category given the set of perturbed samples Xi, λ is determined by maximizing the validation accuracy.

1. Optimal Ensembled Hinge Loss

Here is the objective function:



where

The intuition is that the ensemble output response corresponding to ground truth should be larger than others with a margin γ. With the hinge loss, any case where the response difference is larger than γ will not introduce any penalty. Again, both γ and λ are determined with respect to the accuracy on validation set.

References:

Facial Emotion Detection Using Convolutional Neural Networks and Representational Autoencoder Units Prudhvi Rai Dachapally

Deep Learning for Emotion Recognition on Small Datasets Using Transfer Learning Hong-Wei Ng, Viet Dung Nguyen, Vassilios Vonikakis, Stefan Winkler

Image based Static Facial Expression Recognition with Multiple Deep Network Learning Zhiding Yu Cha Zhang

https://blog.csdn.net/yjl9122/article/details/70198357?utm\_source=blogxgwz2 Convolutional neural networks - input layer, convolutional layer, activation function, pooling layer, fully connected layer

https://blog.csdn.net/JIEJINQUANIL/article/details/50042791

Understanding of pooling in CNN