Detect and Transmit Emotions in Online Chat using Affective Computing

Hui-Po Wang, Fang-Yu Shih
Department of Computer Science, National Tsing Hua University, Hsin Chu, Taiwan

Abstract

We proposed a prototype system which combines spoken content and users' facial expression to generate a corresponding sentence with appropriate emoticons. We also introduce a light-weight CNN model based on VGG-16. The model consists of total 9 CNN layers, 3 maxpooling layers, and 3 fully-connected layers. Although our model is less shallow than VGG-16, our system still has a satisfying performance due the prediction category we adapt. We also analyze the performance and the main reason of why the emotion detection model cannot make a precise prediction in some situation. In the end, we put this project to github in order to make it more extensible.

1. Introduction

With the increasing popularity rate of network user, online chatting is now an indispensable part of our life. Instead of meeting someone face-to-face, we chat online for business, for entertainment, and even for education. Online chatting is based on sending instant message from sender to receiver, made people able to communicate with others immediately. Early online chatting was sending primarily plain text, as time goes by, we can send various type of messages, such as image, audio, and video. Nowadays, once we mentioned online chat[1], emoticon and sticker came to our mind.

The main difference between face-to-face chatting and online chatting is we cannot see each other, which means that instead of observing others mood by their facial expression and speaking tone, we can only deduce others emotion from the words they spoken. And it is really hard to know other's emotion barely from words, so that is why we need emoticons and stickers, to show others how we feel. The advantages of using online chatting instead of audio chatting (e.g. telephone call) and video chatting are: (i) requires less bandwidth, (ii) the content the receiver receives would not be lost due to the instability of the Internet, and (iii) sender and receiver would not be limited by the occasion they are in (e.g. in a meeting or in a lecture). But talking is more convenience than typing, and hence, combining the above mentioned advantages, we design a system that takes facial images (or video) as input, and outputs the corresponding sticker according to the emotion of the sender.

We introduce a new application for online chatting and implement a prototype system combining spoken content and facial expression to generate corresponding plain text content with appropriate emoticons. We use OpenCV [7], which is a powerful open source computer vision library, to detect the region of a human face from the webcam streaming. Then, we crop the region of the face, convert it into gray scale image, and take the image as the input of our pre-trained CNN model to predict user's emotion. Also, we apply Google Cloud Speech API [5] to transform user's spoken content into plain text. We finally combine two results from above to achieve the goal of our application. Moreover, we upload this project to github (https://github.com/a514514772/Real-Time-Facial-Expression-Recognition-with-DeepLearning) and hope that it could be widely used.

2. Related Work

Rowley et al. 1998 [9] uses a neural network model to detect upright frontal human face and apply bootstrap algorithm to rise accuracy of face detection. Instead of using different size window, we use CNN layers to extract features. Viola et al. [13, 12] introduces a ?cascade? machine learning method to rapidly detect objects. It is also implemented by OpenCV [7] and we use the built-in function of the library to detect the region of human face. Krizhevsky et al. [6] shows the capability of neural network in the field of object detection and make neural network research explosively develop. Simonyan et al. [10] show a deeper and more powerful neural network for object detection. The model they proposed is one of the most popular model in the world. Our model is also created based on it.

3. Architecture

In this section, we mainly talk about how we solve the problem and some technical issues we encountered. In 3.1, we first explain the main idea of our solutions, including the

[1]Online chat mentioned throughout the article only consider plain text and image.

system architecture and the workflow. In 3.2, we point out more technical details and explain the design of our model.

3.1. Overview

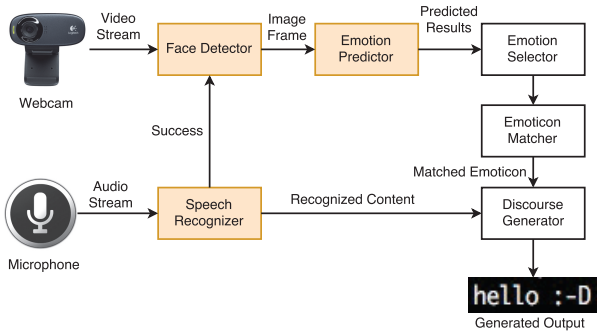

Figure 1. The overview of our proposed system.

Fig. 1 is the architecture of our system. First, we will get the audio stream from the microphone as input, then our **speech recognizer** will outputs the recognized content. If the content is recognized successfully, **face detector** extract some image frames from the video stream captured by the webcam, then crop the facial part of the image and transform it into gray value image. Take the image we preprocessed as input, **emotion predictor** outputs the predicted results of each image. **Emotion selector** selects the most representative emotion according to the probability predicted by the emotion predictor. **Emoticon matcher** matches the emoticons corresponding to the input emotion. Last, **discourse generator** combines the recognized content with the matched emoticon, and then generates the output.

In the following section, we will talk about some of the important components in our system, including face detector, emotion predictor, and speech recognizer, and how to combine the components into ideal results.

3.2. Components

In this section, we will introduce the implementation details of the three components, including face detector, emotion predictor, and speech recognizer (shown in fig. 1). Also, we will go through our workflow.

Speech recognizer transform user?s spoken content into plain text. To achieve this goal, we utilize *SpeechRecognition* [8], a python package, to recognize the audio message. This package include various speech recognition APIs, such as CMU Sphinx [2], Google Cloud Speech API [5], and Bing Speech API [1]. We choose Google Cloud API because it is a built-in function on Chrome, so it is easy for people to use. Google Cloud API enable developers to convert audio into plain text by applying powerful neural network models in an easy-to-use API. At the beginning, our system would try to capture user's voice; if succeed, it transforms the spoken content into plain text and open the Webcam to detect user's facial expression.

The most import work in the **face detector** part is to prepare inputs for the emotion predictor. The quality of the input images decide the accuracy of emotion prediction heavily, so we take advantage of some preprocess skills to increase the quality of the inputs, and process the webcam stream in the frame-level. When a frame comes into the face detector, our system first apply Haar feature-based cascade classifiers [12] to detect the region of the human face, which OpenCV has already implemented as a built-in function. The region detected by OpenCV will be cropped from the original frame, and then we will transform it into gray scale. To prevent the poor conditioning mentioned in a deep learning book [3], we also apply histogram equalization to the gray scale image before sending to the emotion predictor.

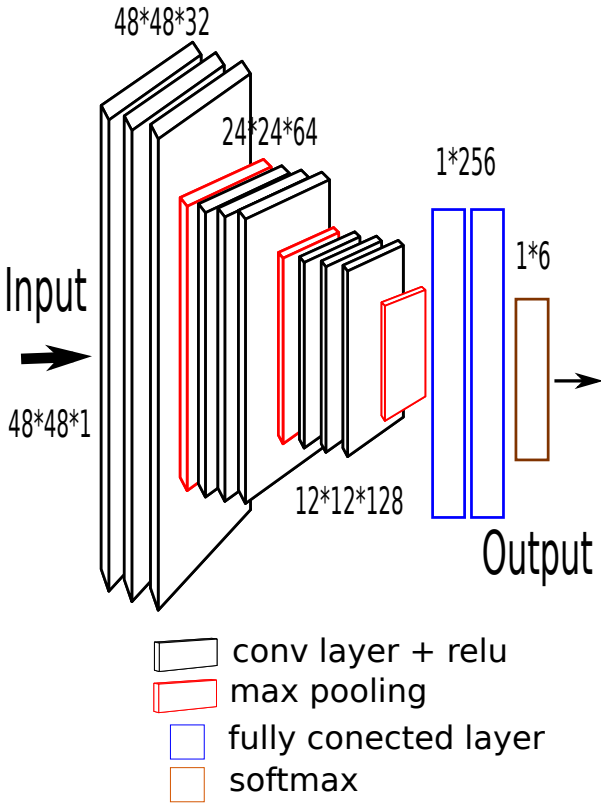

Figure 2. The architecture of our CNN model.

Emotion predictor is another significant module in our system. To accurately predict which type the user's emotion is, we utilize deep learning neural network. Unlike SVM [11] or other conventional machine learning methods, neural network has good anti-noise resistance and enough model complexity for large scale dataset. To recognize hu-

man facial expression, the variety of human's face is one of the most concerned issues in our system. For example, people wearing glasses or the appearance of different races may challenge our pre-trained model on the accuracy of prediction. Therefore, we use neural networks, which can endure more noise to detect various facial expression. In the field of computer vision, there is a special type of neural network called convolutional neural network (CNN). CNN can effectively reduce the number of parameters in the model, and extract locally invariant features from the input image, so we choose CNN as our emotion predictor. We construct our own CNN model based on VGG-16 [10], but we decrease and adjust several layers due to the property of the problem we want to solve and the limitation of hardware. Fig. 2 shows the architecture of our CNN model. The input image of our model is a gray scale image with 48*48 pixels, which is a preprocessed image from the previous module. The CNN part can be considered as several stages. Same as the original design of VGG-16, there are several CNN layers with the filter size 3*3 and increasing number of filters in our model, but we only keep totally 9 CNN layers because the number of the classes we want to predict is only 6 and we don't have enough dataset to prevent overfitting. After every 3 CNN layers, the max-pooling layer with size 2*2 follow and down sample the feature maps which came from the last layer of every 3 CNN layers, and keeps the maximum value in a window of size 2*2. Thus, we can reduce more parameters in the model. The last two layer of the hidden layers are fully-connected layers responsible for extracting more high-level features from the feature maps sent from the CNN layers. In the output layer, the activation function, softmax, normalizes and maps the weight to six values in [0, 1] to represent the probability of six types of emotions respectively.

Above all, we already have a sequence of probability lists which are the results of the prediction corresponding to a sequence of frames, and each list contains six probability values representing six type of emotions respectively. Moreover, we also have the result from speech recognizer which already transformed user's voice into plain text. At the final step, we acquire the most relative emoticon voted by choosing the emotion which has the highest value averaged from the maximum probability value of 5 frames which are captured after our system succeeds in transforming user's voice into plain text. In the end, we append the appropriate emoticon to the end of the sentence produced by speech recognizer to generate the promising result.

4. Experiments

In this section, we'll talk about the accuracy of our model and analyze the results. Here is our hardware information: our CPU is intel i5-6500, and the GPU is Nvidia GTX 960 2G. We also have 8G memory and run our experiments on Ubuntu 16.04.

	precision	recall	f1-score	support
angry	0.48	0.52	0.50	523
fear	0.39	0.30	0.34	496
happy	0.76	0.78	0.81	895
sad	0.44	0.43	0.43	653
surprise	0.70	0.67	0.69	415
neutral	0.48	0.50	0.49	607
avg/total	0.55	0.56	0.56	3589

Table 1. The result of our CNN model.

The dataset we use is fer2013 [4], which consists of 48*48 pixels grayscale images and 28709 training images, 3589 public testing images, and 3589 privacy testing images. The cost function of our model is a categorical entropy. In the training set, our model can achieve 72% accuracy, but we get 56% and 57% accuracy in public testing set and private testing set respectively. It is not satisfying, but in our application, it can be solved by voting from the results of a sequence of frames. We try to understand and analyze why we cannot get higher scores, so we plot out more information in Table 1. We can see that our model cannot distinguish the correct user's emotion when they are fear, sad, or neutral. In more details, we think people show their emotion in different ways, so this leads low prediction value to our model, especially when users are sad or fear. For example, some people may be crying but looks like smiling, and that's why the model cannot predict the emotion so accurately. As to neutral, it is significantly influenced by people's appearance. For example, some people may seem angry if they don't talk to other people. We think the best solution to solve this problem is to collect more and various data including different races, different genders, and even different angles of a person.

In fig. 3, we show more advanced information. Like what Table 1 shows, "fear" and "sad" does not perform well. Also, we can see that the recall value when users are afraid is quite low and the confusion matrix shows that "fear" is usually misclassified to other emotions. It proves our previous assumption that people express their emotion in different ways would lead to a great influence on the prediction. Moreover, a lot of "neutral" instances are misclassified to "sad" label. As we mentioned before, people's appearance significantly influence the prediction. The model cannot perfectly define what the neutral appearance looks like, because people show their emotions in different ways in the training set.

5. Conclusion and Future Work

We designed, implemented, and evaluated our proposed system, which takes facial images (or video) as input, and

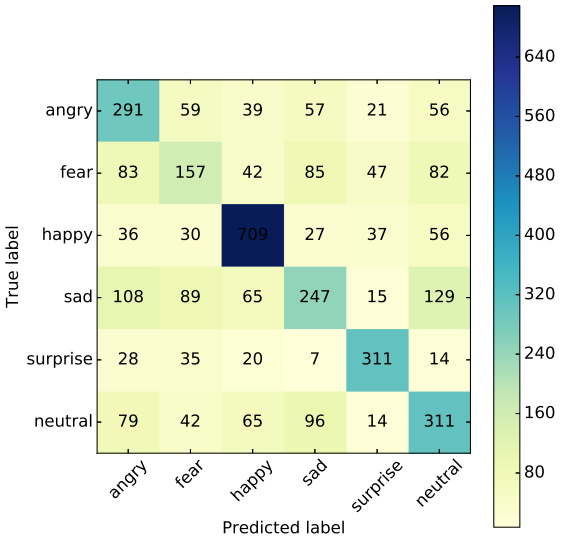

Figure 3. The confusion matrix of our experimets.

outputs the corresponding sticker according to the emotion of the sender. In this paper, we proposed an application, which allow users to chat with people who cannot take advantage of video chatting. Our system combines users' spoken content and facial expression to generate the corresponding sentence with appropriate emoticons. Thus, users can chat with other people in plain text without typing any word. In section 4, we analyze in which condition the performance of our model would change. Also, we make some assumptions to explain the variation of our model, and prove that people show their emotions in different ways may heavily influence the prediction. Moreover, we implement a light-weight model based on VGG-16 to predict users' emotions, and we put our source code to github (https://github.com/a514514772/Real-Time-Facial-Expression-Recognition-with-DeepLearning) to make this project more extensible. In our project, it is acceptable to replace default model with the model which is define by the users.

Our system can be extended in several ways. First, instead of only determine emotion from facial expressions, we will take the tone of the speech content into consideration. Second, we will improve the CNN model to achieve higher accuracy. As we mentioned above, our system cannot reach high accuracy at some emotions (e.g. sad and fear), so we will try to reform our model in order to reach higher accuracy. Last, the emotion of a person is far more than the 6 types we've mentioned (i.e. angry, fear, happy, sad, surprise, and neutral), for example, we can divide "happy" into "smiling" and "laughing". Our system will support more kind of emotions to truly reflect the emotion of the user.

References

[1] Bing Speech API official site. https://www.microsoft.com/cognitive-services/en-us/speech-api.

[2] CMU Sphinx official site. http://cmusphinx.sourceforge.net/.

[3] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[4] I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Cour-ville, 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*, pages 117–124. Springer, 2013.

[5] Google Cloud Speech API official site. https://cloud.google.com/speech/.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[7] Opencv official site. http://opencv.org/.

[8] Python Speech Recognition official site. https://pypi.python.org/pypi/SpeechRecognition/.

[9] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):23–38, 1998.

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

[11] V. Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[12] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[13] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.