Research Proposal

The effect on Linkedin profile picture on the chance to work for Apple

Thanakij Wanavit

1. Introduction

Human beings are a social animal who relies on interpersonal interaction for success as a species. For the reason stated, it is not difficult to see how each individual look can have a significant impact on their opportunities, wellbeing, and how they behave in life in general.

In the early 21st century, social media has transformed the way people communicate and how the whole social interaction is defined.(Deshpande, Deshpande, & O'Brien, 2019) It is fair to say that the social media profile of each individual represents a significant proportion of their mental and physical identity, obtaining the relationship data will be both highly beneficial and controversial to business who would like to target each sector of customer psychographically.

1.1 Problem for investigation

The problem is whether an appearance of a Linkedin profile picture affects a person's chance to work for Apple in the Bay area San Francisco.

1.2 Purpose of the study

The purpose of the study is to understand the reason behind the relationship of person who works for Apple and their appearance.

1.3 Significance of the study

The study has an incredible significance for companies who are looking to either locate their potential employees using pictures, measuring the current employee pool for features bias, or to locate their potential customer segment. A robot image classifier can be of great benefit to almost any related parties.

1.4 Research Question

How does emotions encapsulated in a Linkedin profile picture of a person affect the person's chance to work at Apple in the Bay area San Francisco?

2. Literature review

2.1 The impacts of visuals on human mind

2.1.1 Human history of discrimination

Discrimination is a subject of great exant. Since early in human civilization, they have enslaved, tortured, and discriminated against in many other ways. As late as 1833, it was legal to enslave humans who are visually black. This ended when the English parliment passed Slavery Abolition Act, (1833) which took effect on 1 August 1834. ("Slavery Abolition Act | History & Impact", 2020)

2.1.2 How appearance affects opportunities and success

There are multiple studies on a very broad question of how visual appearance may affect the opportunities. One study by Gatewood, Lahiff, Deter & Hargrove, (1989, 17-31) has found that appearance is the single most important factor in employee selection for a wide variety of jobs.

Various other opportunities such as being in a sports team, music choir, and even a specific school does depend on the appearance of the subject. This, in turn lead to the income discrimination based on apparance. One study by Hamermesh & Biddle (1994,1174-1186) concluded that "plain" people earned five to ten percent less than "average-looking" people, who in turn earned five percent less than "good-looking" people.

Appearance has been found to affect the opportunities and the career path taken with a statistical significance. (Adamitis, 2000, 3-30)

Other than opportunities to learn and find occupation, when getting financing, doing business, and transacting with other people, appearance plays a vital role for obtaining acceptance. The fact that appearance is highly correlated with where a person is from, culture, language, and accent means that they are prone to being stereotyped by others which leads to discrimination conciously or unconciously.

2.2 Profile Picture on Human mind

2.2.1 Profile pictures on different social media outlets

In the modern world, social media has obtained a level of importance beyond just simple personal relationships. Different social media outlet has been build for a specific purpose for instance Linkedin is optimized for sharing professional profile to potential employers and colleague, github is build for sharing coding work with other developers or potential employees, twitter is for sharing opinions to the public, and stackoverflow is for sharing problems and solution to programming problems. Individuals usually post a very different kind of profile picture on each social media platform.

It has been found that most profiles are carefully manipulated for a desirable self-presentation in order to achieve a specific goal.(Larrimore, Jiang, Larrimore, Markowitz & Gorski, 2011,22) Popular presses has linked profiles of those in dating sites to loan sites (Frier, 2009, 6b; Mogul, 2007, 8; Quinn, 2008)

2.2.2 Psychographic Self representation

It has been found that most social media profiles are carefully manipulated for a desirable self-presentation in order to achieve a specific goal.(Larrimore, Jiang, Larrimore, Markowitz & Gorski, 2011,22)

2.3 Image Classification Techniques

2.3.1 Object detection

Yolo real time object detection

You only look once (YOLO) is a system for detecting objects on the Pascal VOC 2012 dataset. It can detect the 20 Pascal object classes. This can be used to separate the picture of human image and image features from the background. (Redmon, et al., 2016)

2.3.2 Features Classification

2.3.2.1 Face features classification

Pose-invariant face recognition can be divided into 4 groups: pose-robust feature extraction, multi-view subspace learning, face synthesis based on 2D methods, adn face synthesis based on 3D methods. One of the way to extract feature is the symmetric interpolation for self-occluded regions (Méndez N, Bouza LA, 2017)

2.3.2.2 Classification with neural network

For each face recognition method, each feature is generated programmatically and given a score. Collectively, these features should summarise all the useful information in the image while reducing the amount of data fed into the final layer of image classifier, usually a fully connected function with softmax activation function(figure 2.3.2.1) which outputs a probability of a classification result. In Simonyan, K., & Zisserman, A. (2015) 's case, it represents the probability of the object containing each of the 1000 pre-trained categories including dog breeds.(K., & Zisserman, A. 2015, p 13). Although these 1000 pre-trained category is only useful for classifying the 1000 classes, a transfer learning method can be used to classify different images into different features as demonstrated by Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014).

$$\mathrm{softmax}(z)_i = \frac{\exp(z_i)}{\sum_{l=1}^{k} \exp(z_l)}$$

Figure 2.3.2.1 Softmax activation function (Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. 2011)

2.3.2.3 Visualising Neural network layers in an image

The method outlined by Graetz, F. M. (2019, July 21) can be used to visualize and plot the locations of the image that has an effect on output feature node of interest(figure 2.3.2.2). This allows us to give it a human description of the node, for example mouth, forehead, hair color.

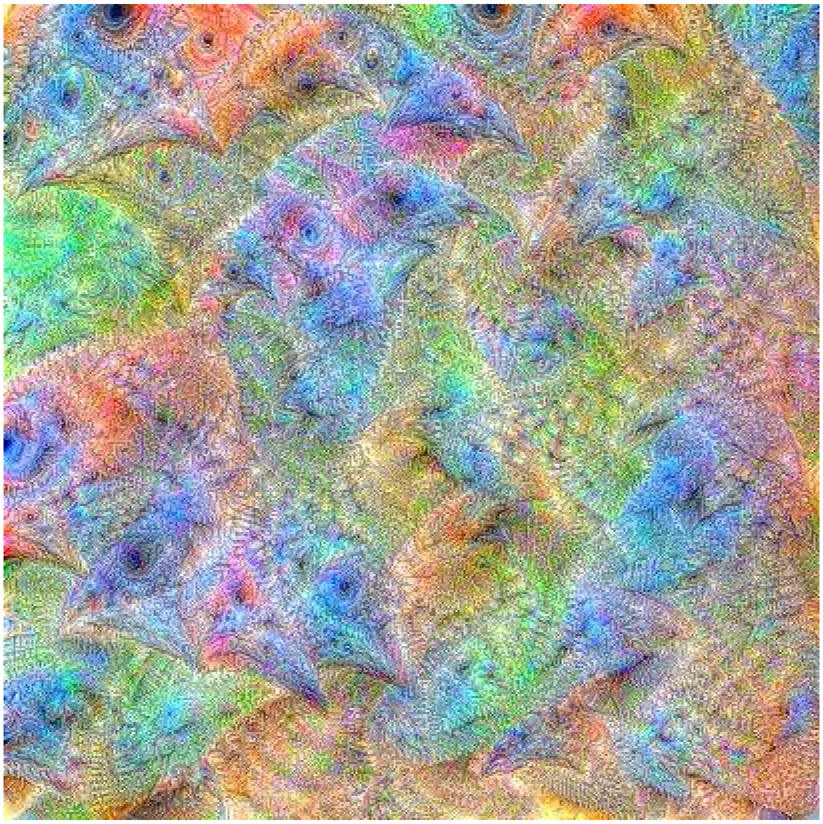

figure 2.3.2.2 Plot of an eye neural network node (Graetz, F. M. 2019, July 21)

2.3.3 Extraction of emotional features from face image

Extraction of emotional features from a face is possible by training a neural network model such as Convolutional Neural Network (Goodfellow, I., Bengio, Y., & Courville, A. 2017) which has proven to be very successful at forecasting emotion (Perez, A. 2018).

Different configurations and variation of Convolutional Neural Network has been tested by researchers to extract the data from the images. These results, although usually highly specific for one set of prediction, e.g. determining if an image contains a specific species of dog, can be transferred to other models with completely different goals using the transfer learning technique (Yang, Zhang, Dai, & Pan, 2020).

One of the model available in EmoPy which is pre-trained is the FER model(Perez, A. 2018), which is a convolutional network trained on the FER dataset (Barsoum, E., Zhang, C., Ferrer, C. C., & Zhang, Z. 2016)

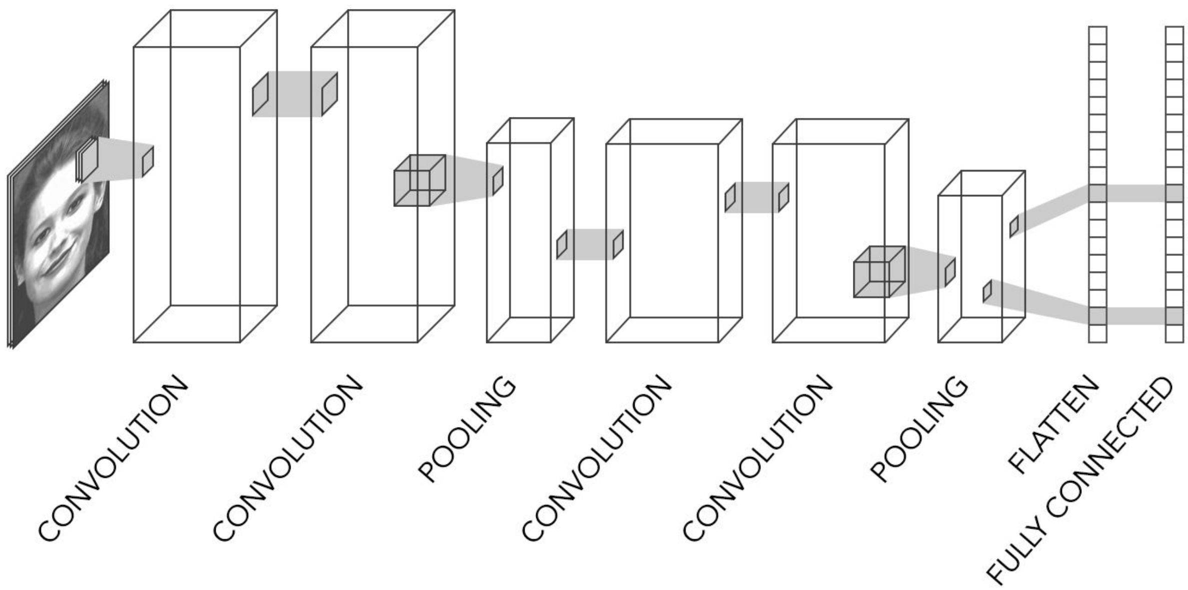

Figure 2.3.3.1 Convolutional Neural network (Perez, A. 2018)

2.3.4 Training data for models

There are many different datasets available publicly to train on a research and competition platform such as Kaggle(Kaggle 2020). Many datasets are geared towards object detection such as OpenImage 2019(Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., & Krasin, I. 2017) . One dataset which is focused on emotional inference is called FER+ dataset (Barsoum, E., Zhang, C., Ferrer, C. C., & Zhang, Z. 2016).

2.3.4.1 FER Dataset

The FER+ annotations provide a set of new labels for the standard Emotion FER dataset. In FER+, each image has been labeled by 10 crowd-sourced taggers, which provide better quality ground truth for still image emotion than the original FER labels. Having 10 taggers for each image enables researchers to estimate an emotion probability distribution per face. This allows constructing algorithms that produce statistical distributions or multi-label outputs instead of the conventional single-label output (Barsoum, E., Zhang, C., Ferrer, C. C., & Zhang, Z. 2016)

3. Methodology Design

The study aims to collect the data between occupation and face features of each male from linkedin in the region of SanFrancisco with the company they work for.

3.1 Methods

3.1.1 Obtain the data from website

- Use browser "Google Chrome" with a webdriver of "chromedriver"
- From the website www.linkedin.com, navigate to search for people page as shown in Fig 3.1.1
- Select current companies from the list of 20 most common companies (fig 3.1.1.2)
- Save this picture and company name in a database
- Collect 100 samples from each company

A video demo of the following task can be viewed at
https://www.youtube.com/watch?v=N3sfYo867Sk

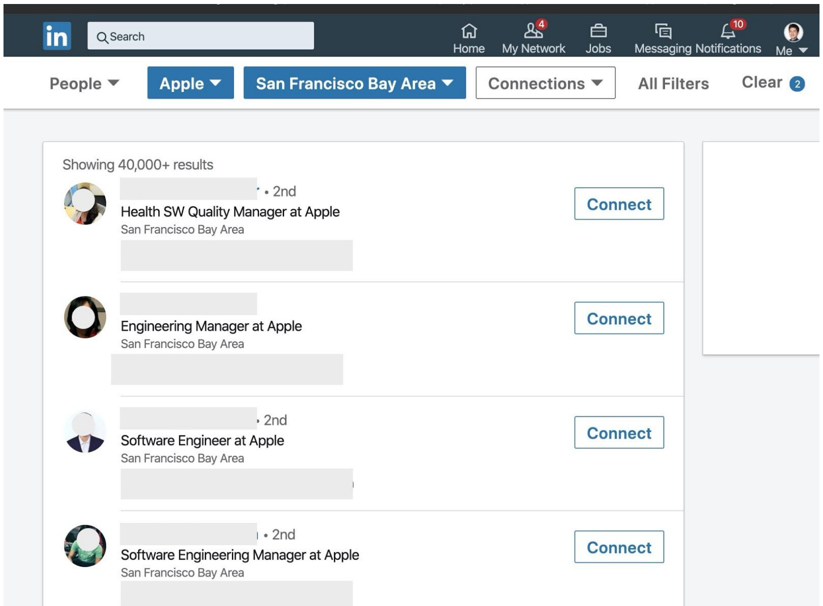

Fig 3.1.1 Search for people page (www.linkedin.com, 10 Feb 2020)

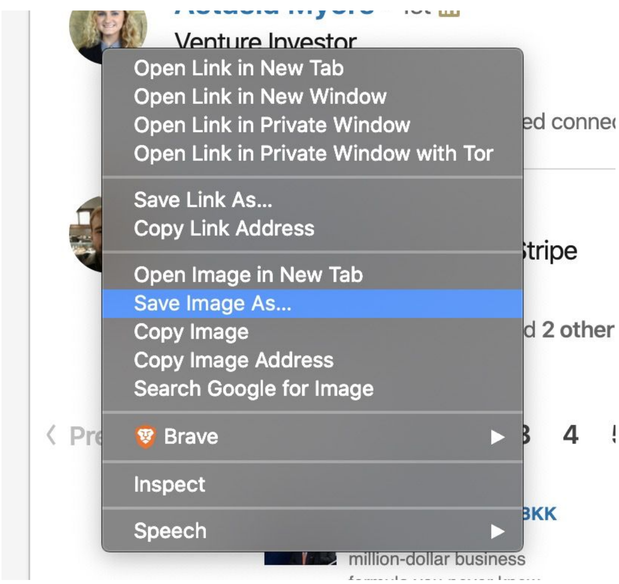

Fig 3.1.1.2 save image from profile picture within the linkedin search page (www.linkedin.com, 10 Feb 2020)

3.1.2 Extract emotional feature from each face

- Using a package management tool "pip"

- Install EmoPy (Perez, A.,2017) and its prerequisite from the official repository
- Import pre-trained FER model (Barsoum, E., Zhang, C., Ferrer, C. C., & Zhang, Z. ,2016)
- Obtain probabilistic prediction of the following emotion attributes;

```
['calm', 'anger', 'happiness', 'surprise', 'disgust', 'fear', 'sadness']
```

Figure 3.1.2.1 list of emotions of interest

- Scores are recorded for each picture
- Source code for the task can be found in appendix (7.3.1.2)

A video demo of the following task can be viewed at *https://www.youtube.com/watch?v=PFrGgnHgdRQ*

3.1.3 Features analysis

- The test data accuracy should represent the degree of pattern correlation between each company
- For each feature generated, plot and output of the generated feature, select the top feature for each company and take an average
- Generate a human description of each feature (eg forehead, chin, mouth) depending on the plot
- This should indicate which part of the profile image feature is important in determining career

3.2 Sampling

- Systematic sampling using 5 separate robots with an account in oregon to collect 100 pictures randomly.
- Sample size is 200 Pictures

3.3 Instrumentation

- Data will be collected using selenium(SeleniumHQ, 2019) controller which controls google chrome browser on a cloud server using aws ec2 instance t3.micro.
- Data collected will be uploaded to s3 bucket for storage and the uri will be collected in a dynamodb database.(Rangel, 2015)

3.4 Data Analysis

- Score on each emotion listed in Figure 3.1.2.1 are obtained and put into a single database
- The score for each emotional feature in each group (Apple or others) is analyzed for standard deviation, covariance, average.
- The most important face feature is the one which has the highest standard deviation when comparing employees from different companies, but low standard deviation when comparing the employees from the same company
- The final outcome will be represented as heatmap of correlation between features and companies as in fig 3.4.1

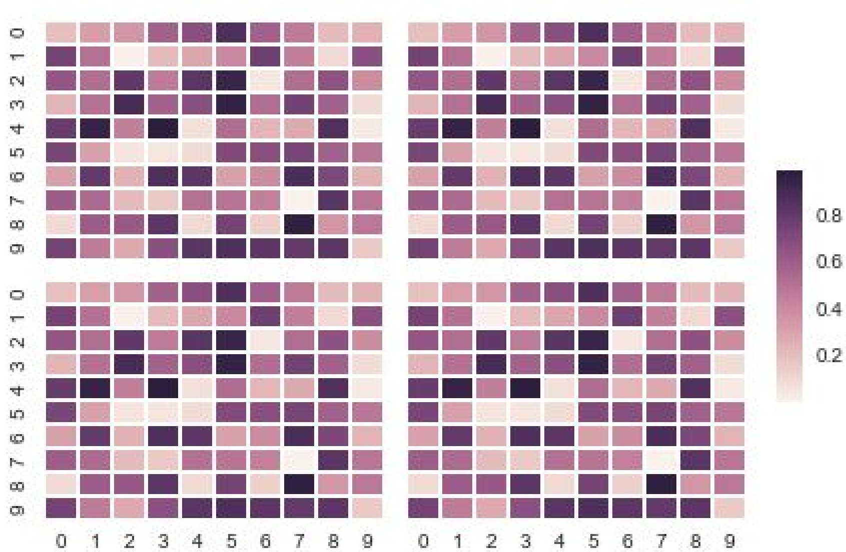

Fig 3.4.1 Generic heatmap diagram generated with seaborn 0.10.0 (https://seaborn.pydata.org/, 10 Feb 2020)

4. Limitations and Delimitations

4.1 Limitations

There are many social media which are designed for different purposes. Facebook is for sharing with friends and family, Linkedin is for sharing with potential employees and colleagues, Twitter is for sharing with the public, and Instagram is for sharing only pictures with followers. Pictures taken from different sources, platforms will give a different relationship

It is difficult to sample fairly due to the fact that there is a machine learning which ranks which profile to show to your account first which leads to an algorithmic bias.

4.2 Delimitations

Pictures should be taken from the linkedin profile of people who are located in the USA and have opened their profile to the public.

To reduce bias, Use a robot to login from multiple locations and create 5 linkedin profiles to do so. Hide the ip and use an Oregon based ip to interface with linkedin

5. Significance of the study

The study will allow companies to create a tool to make a better decision based on how they select candidates. Biases racially, emotionally, and morally.

For candidates who would like to apply for jobs, this method may give an insight to the chance of being accepted at a specific company.

6. References

Adamitis, E. (2000). Appearance Matters: A Proposal to Prohibit Appearance Discrimination in Employment. Washington Law Review, 75(1), 3-30.

Barsoum, E., Zhang, C., Ferrer, C. C., & Zhang, Z. (2016). Training deep networks for facial expression recognition with crowd-sourced label distribution. ICMI 2016 - Proceedings of the 18th ACM International Conference on Multimodal Interaction, 279–283. https://doi.org/10.1145/2993148.2993165

Brittanica (2020) Slavery Abolition Act | History & Impact.. Retrieved 26 January 2020, from https://www.britannica.com/topic/Slavery-Abolition-Act

Deshpande, A. K., Deshpande, S. B., & O'Brien, C. A. (2019). Hyperacusis and social media trends. Hearing, Balance and Communication, 17(1), 1–11. doi: 10.1080/21695717.2018.1539321
Frier, S.(2009, June 30). Peer-to-peer lending is flourishing amid the credit crisis. Sacremento Bee,p. 6b.

Gatewood, R., Lahiff, J., Deter, R., & Hargrove, L. (1989). Effects Of Training On Behaviors of The Selection Interview. Journal Of Business Communication, 26(1), 17-31. doi: 10.1177/002194368902600103

Goodfellow, I., Bengio, Y., & Courville, A. (2017). Deep learning. Cambridge, MA: The MIT Press.

Guido, G., Pichierri, M., Pino, G., & Nataraajan, R. (2018). Effects of Face Images and Face Pareidolia on Consumers Responses to Print Advertising. Journal of Advertising Research, 59(2), 219–231. doi: 10.2501/jar-2018-030

Hamermesh, D., & Biddle, J. (1994). Beauty and the Labor Market. American Economic Review, dec, 1174-1186.

Kaggle (2020).,Your Machine Learning and Data Science Community, Retrieved March 8, 2020, from https://www.kaggle.com/

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., & Krasin, I. (2017). The Open Images Dataset V4 Unified image classification , object detection , and visual relationship detection at scale.
Perez, A. (2018). EmoPy: a machine learning toolkit for emotional expression. ThoughtWorks. https://www.thoughtworks.com/insights/blog/emopy-machine-learning-toolkit-emotional-expression

Larrimore, L., Jiang, L., Larrimore, J., Markowitz, D., & Gorski, S. (2011). Peer to Peer Lending: The Relationship Between Language Features, Trustworthiness, and Persuasion Success. Journal Of Applied Communication Research, 39(1), 19-37. doi: 10.1080/00909882.2010.536844

Méndez N, Bouza LA, Chang L, Méndez-Vázquez H, editors. Efficient and Effective Face Frontalization for Face Recognition in the Wild. Iberoamerican Congress on Pattern Recognition; 2017: Springer.

Mogul, M. (2007, July). Look for a higher return? Consider lending to a peer. Kiplinger Business Forecasts, 8. http://www.kiplinger.com/businessresource/forecast/archive/consider_lending_ to_a_peer_070709.html.

Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830

Perez, A. (2017). EmoPy (0.0.5). ThoughtWorks Arts. https://github.com/thoughtworksarts/EmoPy

Perez, A. (2018). EmoPy: a machine learning toolkit for emotional expression. ThoughtWorks. https://www.thoughtworks.com/insights/blog/emopy-machine-learning-toolkit-emotional-expression

Quinn, J. (2008, August 27). Financing is as close as the nearest angel. Retrieved from http:// www.bloomberg.com/apps/news?pid=20601039&sid=a7LI2Jh3XdHM

Rangel, D. (2015). DynamoDB: everything you need to know about Amazon Web Service's NoSQL database. Retrieved from https://aws.amazon.com/dynamodb/

Redmon, Joseph, Divvala, Santosh, Girshick, Ross, … Ali. (2016, May 9). You Only Look Once: Unified, Real-Time Object Detection. Retrieved from https://arxiv.org/abs/1506.02640

SeleniumHQ. (2019, December 10). SeleniumHQ/selenium. Retrieved from https://github.com/SeleniumHQ/selenium

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, 1–14.

Yang, Q., Zhang, Y., Dai, W., & Pan, S. J. (2020). Transfer learning. Cambridge: Cambridge University Press.
Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? Advances in Neural Information Processing Systems, 4(January), 3320–3328.
Graetz, F. M. (2019, July 21). How to visualize convolutional features in 40 lines of code.
Retrieved from https://towardsdatascience.com/how-to-visualize-convolutional-features-in-40-lines-of-code-70b7d87b0030

7.Appendix

7.1 Resnet - 50

NetChain[

		image
	Input	array (size: 3×224×224)
conv1	ConvolutionLayer	array (size: 64×112×112)
bn_conv1	BatchNormalizationLayer	array (size: 64×112×112)
conv1_relu	Ramp	array (size: 64×112×112)
pool1_pad	PaddingLayer	array (size: 64×113×113)
pool1	PoolingLayer	array (size: 64×56×56)
2a	NetGraph (12 nodes)	array (size: 256×56×56)
2b	NetGraph (10 nodes)	array (size: 256×56×56)
2c	NetGraph (10 nodes)	array (size: 256×56×56)
3a	NetGraph (12 nodes)	array (size: 512×28×28)
3b	NetGraph (10 nodes)	array (size: 512×28×28)
3c	NetGraph (10 nodes)	array (size: 512×28×28)
3d	NetGraph (10 nodes)	array (size: 512×28×28)
4a	NetGraph (12 nodes)	array (size: 1024×14×14)
4b	NetGraph (10 nodes)	array (size: 1024×14×14)
4c	NetGraph (10 nodes)	array (size: 1024×14×14)
4d	NetGraph (10 nodes)	array (size: 1024×14×14)
4e	NetGraph (10 nodes)	array (size: 1024×14×14)
4f	NetGraph (10 nodes)	array (size: 1024×14×14)
5a	NetGraph (12 nodes)	array (size: 2048×7×7)
5b	NetGraph (10 nodes)	array (size: 2048×7×7)
5c	NetGraph (10 nodes)	array (size: 2048×7×7)
pool5	PoolingLayer	array (size: 2048×1×1)
flatten_0	FlattenLayer	vector (size: 2048)
fc1000	LinearLayer	vector (size: 1000)
prob	SoftmaxLayer	vector (size: 1000)
	Output	class

]

https://github.com/WeidiXie/Keras-VGGFace2-ResNet50

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

7.3.1.2 Source code for giving scores to emotions in images

```python
from keras.models import load_model
import cv2
from scipy import misc
import numpy as np
import json
from pkg_resources import resource_filename
import imageio

class FERModel:
    """
    Pretrained deep learning model for facial expression recognition.
    :param target_emotions: set of target emotions to classify
    :param verbose: if true, will print out extra process information
    **Example**::
        from fermodel import FERModel
        target_emotions = ['happiness', 'disgust', 'surprise']
        model = FERModel(target_emotions, verbose=True)
    """

    POSSIBLE_EMOTIONS = ['anger', 'fear', 'calm', 'sadness', 'happiness', 'surprise', 'disgust']

    def __init__(self, target_emotions, verbose=False):
        self.target_emotions = target_emotions
        self.emotion_index_map = {
            'anger': 0,
            'disgust': 1,
            'fear': 2,
            'happiness': 3,
            'sadness': 4,
            'surprise': 5,
            'calm': 6
        }
        self._check_emotion_set_is_supported()
        self.verbose = verbose
        self.target_dimensions = (48, 48)
        self.channels = 1
        self._initialize_model()

    def _initialize_model(self):
        print('Initializing FER model parameters for target emotions: %s' % self.target_emotions)
        self.model, self.emotion_map = self._choose_model_from_target_emotions()

    def predict(self, image_file):
        """
        Predicts discrete emotion for given image.
        :param images: image file (jpg or png format)
        """
        image = imageio.imread(image_file)
        return self.predict_from_ndarray(image)
```

```
    def predict_from_ndarray(self, image_array):
        """
        Predicts discrete emotion for given image.
        :param image_array: a n dimensional array representing an image
        """
        gray_image = image_array
        if len(image_array.shape) > 2:
            gray_image = cv2.cvtColor(image_array, code=cv2.COLOR_BGR2GRAY)
        resized_image = cv2.resize(gray_image, self.target_dimensions, interpolation=cv2.INTER_LINEAR)
        final_image =
np.array([np.array([resized_image]).reshape(list(self.target_dimensions)+[self.channels])])
        prediction = self.model.predict(final_image)
        # Return the dominant expression
        dominant_expression = self._print_prediction(prediction[0])
        return dominant_expression

    def _check_emotion_set_is_supported(self):
        """
        Validates set of user-supplied target emotions.
        """
        supported_emotion_subsets = [
            set(['calm', 'anger', 'happiness', 'surprise', 'disgust', 'fear', 'sadness']),
            set(['anger', 'fear', 'surprise', 'calm']),
            set(['happiness', 'disgust', 'surprise']),
            set(['anger', 'fear', 'surprise']),
            set(['anger', 'fear', 'calm']),
            set(['anger', 'happiness', 'calm']),
            set(['anger', 'fear', 'disgust']),
            set(['calm', 'disgust', 'surprise']),
            set(['sadness', 'disgust', 'surprise']),
            set(['anger', 'happiness'])
        ]
        if not set(self.target_emotions) in supported_emotion_subsets:
            error_string = 'Target emotions must be a supported subset. '
            error_string += 'Choose from one of the following emotion subset: \n'
            possible_subset_string = ''
            for emotion_set in supported_emotion_subsets:
                possible_subset_string += ', '.join(emotion_set)
                possible_subset_string += '\n'
            error_string += possible_subset_string
            raise ValueError(error_string)

    def _choose_model_from_target_emotions(self):
        """
        Initializes pre-trained deep learning model for the set of target emotions supplied by user.
        """
        model_indices = [self.emotion_index_map[emotion] for emotion in self.target_emotions]
        sorted_indices = [str(idx) for idx in sorted(model_indices)]
        model_suffix = ''.join(sorted_indices)
        #Modify the path to choose the model file and the emotion map that you want to use
        if(model_suffix == '0123456'):
            model_file = 'models/conv_model_%s.h5' % model_suffix
        else:
            model_file = 'models/conv_model_%s.hdf5' % model_suffix
        emotion_map_file = 'models/conv_emotion_map_%s.json' % model_suffix
        emotion_map = json.loads(open(resource_filename('EmoPy', emotion_map_file)).read())
```

```
        return load_model(resource_filename('EmoPy', model_file)), emotion_map

    def _print_prediction(self, prediction):
        if self.verbose:
            normalized_prediction = [x/sum(prediction) for x in prediction]
            for emotion in self.emotion_map.keys():
                print('%s: %.1f%%' % (emotion, normalized_prediction[self.emotion_map[emotion]]*100))
            dominant_emotion_index = np.argmax(prediction)
            for emotion in self.emotion_map.keys():
                if dominant_emotion_index == self.emotion_map[emotion]:
                    dominant_emotion = emotion
                    break
            # print('Dominant emotion: %s' % dominant_emotion)
            # print()
        else:
            print('verbose is False')
        return prediction
target_emotions = ['anger', 'fear', 'surprise', 'calm']
model = FERModel(target_emotions, verbose=False)

prediction = model.predict('nicpic.jpg')

normalized_prediction = [x/sum(prediction) for x in prediction]

result_dict = {}
for emotion in model.emotion_map.keys():
    # print('%s: %.1f%%' % (emotion, normalized_prediction[model.emotion_map[emotion]]*100))
    result_dict[emotion] = normalized_prediction[model.emotion_map[emotion]]*100
result_dict
```

Appendix 7.3.1.2 Source code for emotions scoring, partially modified version of Emopy, Perez, A. (2017)