# The Utilization of Cloud Computing for Facial Expression Recognition using Amazon Web Services

Gregorius Rafael, Hendra Kusuma, Tasripan
Electrical Engineering Department, Institut Teknologi Sepuluh Nopember, Indonesia
gregorius.rafael16@mhs.ee.its.ac.id, hendraks@ee.its.ac.id, tasripan@ee.its.ac.id

***Abstract* — Face-to-face communication is a form of interaction that are done by every people, but it is hard to be implemented for blind person, especially for them to recognize their interlocutors' facial expression. Hence, a device is needed for recognizing facial expression, so blind people could recognize their interlocutors' facial expression clearly and those expressions converted into non-verbal information in type of sound.**

**This research will use embedded device designed for machine learning called AWS DeepLens, which purposed for looking the performance of the device when using Amazon Web Services (AWS) cloud computing services for AI-trained model and compare it to on-premises deep learning. The device's outputs will inform facial expression information to the user via sound. Deep learning algorithm success will be measured in confusion matrix and its average rate is 76,16%. This proposed device utilizes cloud computing system from AWS, with objectives of system robustness and real-time usage.**

**Keywords—Cloud Computing, Deep Learning, Visually-Impaired, Non-verbal Communication**

## I. INTRODUCTION

On daily life, visually impaired people tend to struggle with many challenges on their lives, such as looking out streets in unfamiliar neighborhoods, detecting objects and people and recognizing faces with expressions. One of the main challenges when visually impaired persons interacts directly face to face is the unrecognized non-verbal information from his/her interlocutor, like facial expressions and body gestures. Facial Expression is believed that it has strong relation with someone's emotion and capable of informing more comprehend information while communicating. Visually impaired people cannot retrieve visual non-verbal information due to their visual limitation. Human can also process regularly received information using visions with another senses, like sound or touch [1]. For some information like colors, writings, non-verbal communication, or body language could be received by hearings or haptic belt. Most common example for this is Braille characters, which commonly used for written information delivery via touch. There has already been a research on delivering visual information for visually impaired people using haptic belt [2]. But the design of that device is less practical due to its big size and its manufacturing costs. These factors make assistive devices less able for commercials. It is necessary to think and create a more practical assistive device with affordable price, so that it could reach more device's users.

Another example for visual information delivery for visually impaired people research is the facial expression recognition device for blind people made using the Raspberry Pi device as a deep learning algorithm processor, with a camera mounted on a hat as an image capture and a simple headset for output from the device. The combination of beep sounds that come out of the headset will become a reference for certain facial expressions or emotions that the system will detect through the camera [3]. This solution is cost-efficient as an assistive technology, but there is no cloud adoption included in this research. A practical, better performance and cost-efficient assistive device will be developed in this paper, so there will be lots of visually impaired assisted with this device. One of the key features that are researched and developed in this paper is visual emotion/expression classifier and its responses with cloud computing adoption.

## II. RELATED WORK

In general, facial expression recognition system that are implemented on an embedded device already exist. For example, Henrik Buimer *et al.* use haptic belts for conveying facial expression information to the visually impaired person [2], Hasby Fahrudin *et al.* implemented a simple single board computer, a webcam and earphones for audio code delivery system [3], or Yongsik Jin *et al.* implemented a cane with vibrotactile device for face emotion detections [4]. In this paper, we will do another approach, compared to those three facial expression recognition assistive device studies. Since those studies did not take any cloud adoptions in it, then we will try a method which involves cloud computing adoption for its deep learning model training and implementation using AWS as the cloud computing services and AWS DeepLens device, which integrates easily with AWS systems.

## III. METHODS

In general, facial expression recognition and identification begins with facial image capture. The captured image will be process to the processing unit. Before the image processed into prediction system, it will be augmented first. The prediction results will be delivered using audio information.

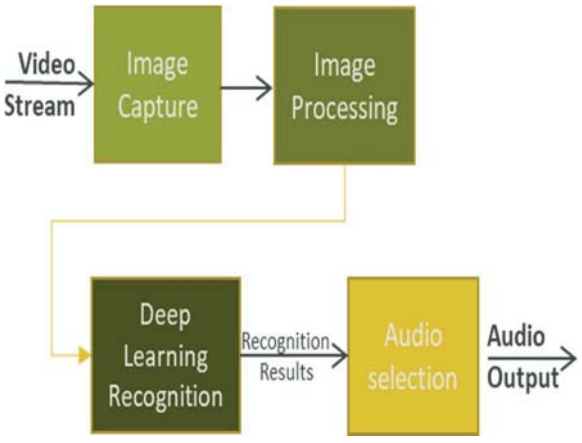



Figure 1. Proposed System Block Diagram

### A. Datasets Preparation

The greatest factor in the quality of deep learning model is the dataset. The more representative the dataset used, the better the model performance. The dataset used in this paper

978-1-7281-8283-4/20/$31.00 ©2020 IEEE

is FER-Plus (FER+) [5,6]. This dataset is needed for deep learning algorithm training which will be implemented into the system in this study.

The images contained in the FER+ dataset have been labeled according to 7 kinds of emotions, namely sad, happy, afraid, disgusted, angry, surprised, and neutral, by saving the images in a folder whose title matches facial expressions. The total number of images in this dataset is approximately 28,000 images.

To ensure the quality of the dataset, we checked the images one by one in each folder. Images with facial expressions that do not match the title of the folder where they are stored will be moved to the appropriate folder. If there is an image of an ambiguous facial expression, the image will be deleted. At this stage, the Amazon SageMaker labeling jobs service is used to classify facial expressions contained in the dataset.

### B. Deep Learning Model and Training

In this paper, a model performance search will be carried out with *learning rate*, *decay*, and *epoch* variables. Important data to be taken is the accuracy and speed of prediction from the model. The CNN architecture used is VGG-16 with pretrained weights [7]. The VGG-16 architecture used is modified in the activation function section, where the activation function used is Elu (Exponential Linear Unit). The choice of VGG16 architecture was based on the availability of a small resolution dataset and because the AWS library also supports the VGG-16 architecture with Keras. Specifically, the VGG-16 architecture can be trained with low-resolution images, thus the training speed is relatively faster and better results can be obtained, although using the Elu activation function is more time consuming for computational training calculations. In order to get better training speed and model prediction, the modified VGG-16 architecture is used in this paper referring to K. Pasupa *et al.* with their new-modified VGG-16 architecture [8] and Z. Yang *et al.* with their convolutional feature [9].

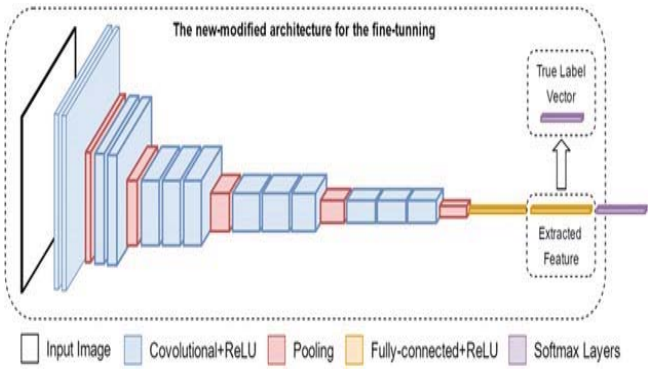



Figure 2. Illustrations of new-modified VGG-16 [8]

The output of this architecture has fairly simple classes, so that the output is immediately used as a reference for the output target. Input from this deep learning model is a color image with a resolution of 48x48 pixels. The image of the deep learning architecture used can be seen in Figure 2. The output of the model training is a weights model with HDF5 (.h5) and json (.json) formats. The output must be converted into a file in the Protocol Buffer format, according to the training output format from TensorFlow and the format supported by the Amazon SageMaker system. After conversion of training results, a way is needed to convert the deep learning model into a trigger so that it can be processed in AWS Lambda. The conversion results are stored in an S3 bucket which has been automatically created.

### C. Prediction Speed

The system proposed in this study uses CNN. CNN is known to require considerable computing power so that it can slow down the work of a system. Therefore, it is necessary to do an experiment to find a model that fits the application in this study. To select the appropriate model, a speed test of 2 CNN architectures was carried out, namely VGG-16, and VGG-16 which were modified in their activation functions. The architecture is trained on a dataset without doing any parameter learning. The test is done by predicting 50 images with respect to the speed and accuracy of the predictions on the test computer and the AWS DeepLens (see TableI).

TABLE I. TEST RESULTS ON TEST COMPUTER AND AWS DEEPLENS

| Architecture | | Speed (seconds/prediction) | Accuracy |
|---|---|---|---|
| **Test Computer** | VGG-16 | 0.05123 | 62 % |
| | Modified VGG-16 | 0,02845 | 70 % |
| **AWS DeepLens** | VGG-16 | 6,218 | 58 % |
| | Modified VGG-16 | 1,362 | 65 % |

TABLE II. TEST COMPUTER SPECIFICATIONS [10]

| Computer Type | ml.c5.2xlarge (*Computing Optimized*) |
|---|---|
| **CPU** | 8 Core |
| **RAM** | 16 GB |
| **Storage Memory** | 5 GB |

On the test computer experiment, the difference in accuracy and speed is not too significant. If the model speed is less than 0.1 second, then the model is very good for predicting continuous images. There is no significant difference in accuracy between the two architectures. However, there is a very significant difference in the speed of the AWS DeepLens. Based on observations, the modified VGG-16 is very superior at prediction speed although it is not yet a good speed. However, in this research application the speed has been fulfilled. In terms of accuracy, the three models have relatively the same speed. Of the two architectures, the modified VGG-16 was the most suitable for use in this study.

### D. Hyperparameter Tuning

In this study, learning parameter settings will be carried out. The parameters to be considered are the maximum epoch rating and learning. In this test, the VGG-16 model will be trained using the FER+ dataset. First, the model will be trained several times with a random learning rate parameter to obtain an optimal learning rate reference value with a maximum epoch limit of 25 epochs. After finding the optimal reference value, a learning model is carried out using the previously found reference value. In Figure 3 with momentum 0.6 and learning rate of 0.001, those can produce an accuracy value of 47.97% and a loss of 1.4%. This can be said to be good, considering that the dataset is only 28,000 samples and is limited to only 11 epochs from the maximum epochs limit of 12 epochs. For more robust results, this estimated value is

needed to achieve its maximum result for learning, which becomes a more robust final model.

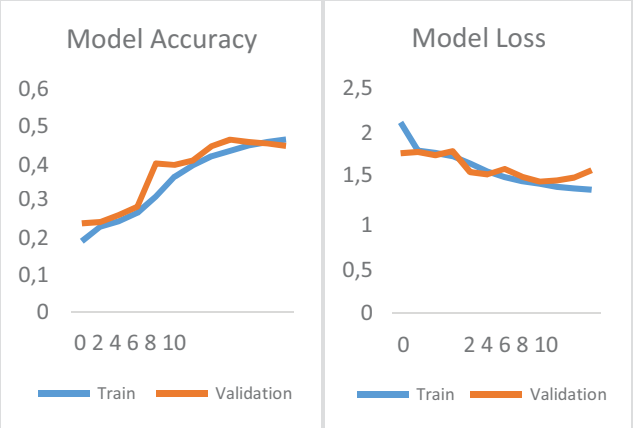



Figure 3. Learning History with Momentum 0.6 and Learning Rate 0.0001.

After getting the reference value for the momentum variable and the learning rate, a repeat learning session will be held. The model from this learning session will be the final model that will be implemented into a software system that is designed when it reaches the minimum required accuracy, which is 50% of the test data. The expectation of hyperparameter tuning is to get a model whose accuracy is more than the minimum limit. Figure 4 is a history of training sessions where hyperparameter tuning has been carried out with the value of the momentum and learning rate variables respectively 0.8 and 0.0002. The accuracy of training data is almost 60%. If we observed it carefully, the accuracy of the test data has exceeded the minimum limit, which is close to the 70% threshold. Therefore, the model from this learning session will be used as the final model to be implemented in the system.

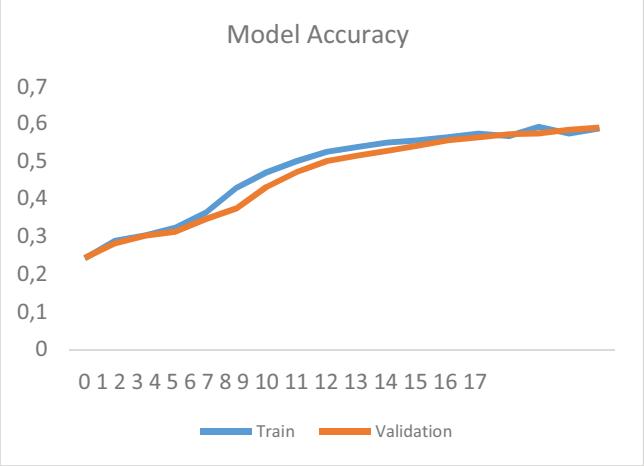



Figure 4. Model Accuracy using Optimal Learning Rate and Momentum.

### *E. Software Design*

For system implementation, it is necessary to create a script with the python programming language version 3.7 on the AWS Lambda service, so that the script created can be called for download on AWS DeepLens devices via the AWS Lambda service. The process of executing so that deep learning models can be instantiated on AWS Lambda is to convert the training results to .pb format located on Amazon S3. Then, after the training results can be converted into triggers for AWS Lambda, Amazon S3 instances can be called upon to program in AWS Lambda. Figure 5 is a flow chart of the script algorithm that will be built on AWS Lambda. The program that will be created will be imported into the AWS DeepLens device automatically by connecting the device with the AWS DeepLens console with the existing AWS Lambda.

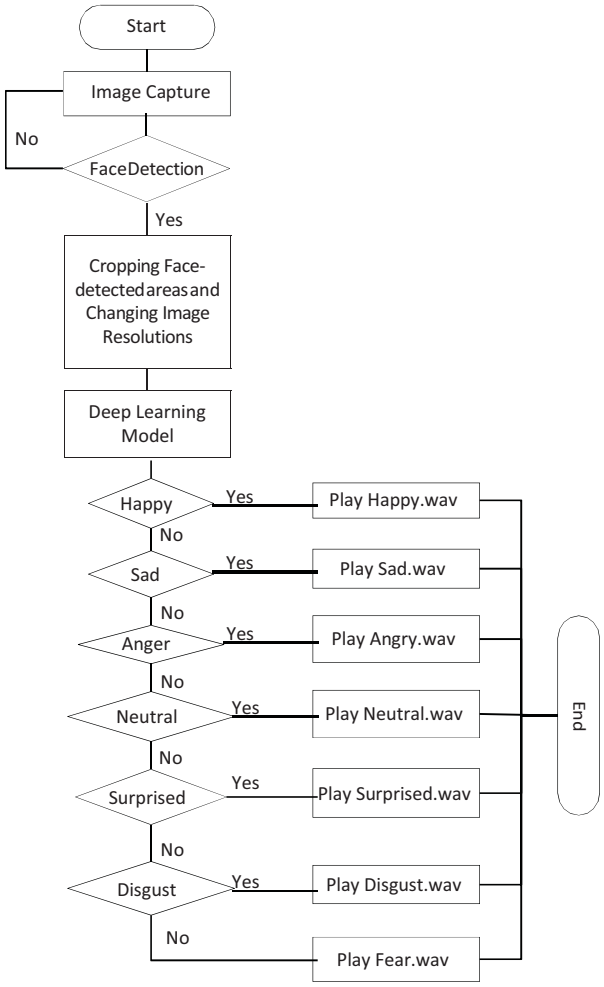



Figure 5. Software Implementation Algorithm on AWS DeepLens Device.

Script will run automatically when AWS DeepLens starts up. Images captured by the camera will be processed in python with the help of OpenCV. OpenCV and Haar-cascade can detect human faces in these images. When detected, the face with the largest area will be taken and filtered by the histogram equalization filter according to the default OpenCV. The resolution is lowered to 48x48 pixels, so that images can enter deep learning models. Each label in the deep learning output will be programmed with a specific voice. The prediction results will be presented in the form of an audio file for each facial expression will be assigned to a different audio file created by recording the sound according to the expression in the form of wav. Instead of using a different tone for each facial expression, using the voice of the expression directly feels more direct without having to memorize a specifically coded tone.

## IV. Results

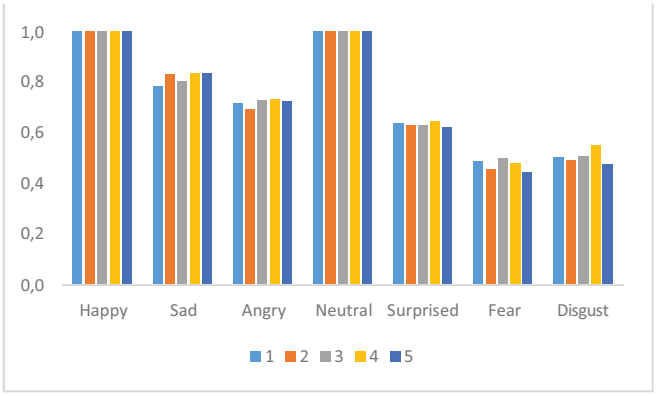
In this study, the device will be tested on two user categories, namely users with normal vision and blind people. This test is expected to find a correlation between the category of device users and the ability to receive voice information.

The test scenario will be considered in order to analyze the test result data comprehensively.

### A. *Device Testings on Users with Normal Vision*

In this test, 5 participants were invited as device users and 3 of them were also expressors. The participant who acts as an expressor will demonstrate 7 facial expressions for 3 times. The user of the device will guess the expression made by expressor.

Before starting the test, participants will be shown sound information on each expression and a picture of facial expressions in the dataset as a reference in plotting each facial expression. Expressors will sit next to each other without eye contact with the device user. The device user will speak the detected facial expression information through the earphones. The percentage of success and error of device users will be logged. Data that has been taken from each user of the device will be put together and presented with a confusion matrix.




| | Happy | Sad | Angry | Neutral | Surprised | Fear | Disgust |
|---|---|---|---|---|---|---|---|
| Happy | 1,00 | - | - | - | - | - | - |
| Sad | - | 0,82 | 0,12 | - | - | - | - |
| Angry | - | 0,11 | 0,73 | 0,03 | 0,13 | - | - |
| Neutral | - | - | - | 1,00 | - | - | - |
| Surprised | 0,13 | - | - | 0,21 | 0,63 | - | 0,05 |
| Fear | - | 0,09 | - | 0,13 | - | 0,54 | 0,24 |
| Disgust | 0,13 | 0,25 | 0,11 | - | - | - | 0,51 |

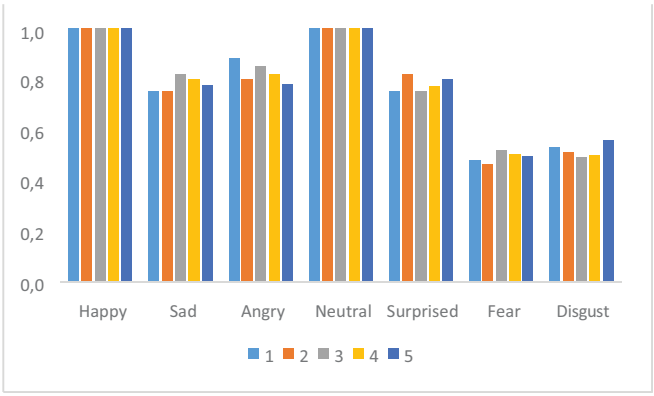
Figure 6. The Results of Normal Vision Users Test (a) Success Rate of each Expression (b) Confusion Matrix of Facial Expression Recognition

In Figure 6a, the prediction success rates of every subjects differ in several categories of expressions. However, each subjects have a similar success trend. Seen in Figure 6b, the happy and neutral expression do not have any prediction. Sad expression has the lowest perdiction error rate, followed by anger, surprise, and fear expression respectively. Disgust has the most prediction errors in this test. For other expressions, there are prediction errors but they are still within reasonable limits. The system has a total prediction accuracy of 75.25% and it can be said that the system is running well.

### B. *Device Testings on Visually Impaired Users*

In this test, 5 blind participants were invited as device users with 3 participants from previous test as expressors. This test is carried out by a blind person as a device user and three persons with normal vision as an object to recognize their facial expressions. Participants will demonstrate 7 facial expressions three times to device users.

Before the test begins, device users are trained to recognize voice information from the earphones for 10 minutes. Participants who play facial expressions will sit next to the user of the device. The display will show some examples of facial expressions in the dataset for reference. The user of the device will guess the display's facial expression based on the sound information heard. The examiner will record every wrong or correct guess in order to present the final data as confusion matrix. After the test, user feedback session is held about the device being tested.




| | Happy | Sad | Angry | Neutral | Surprised | Fear | Disgust |
|---|---|---|---|---|---|---|---|
| Happy | 1,00 | - | - | - | - | - | - |
| Sad | - | 0,78 | 0,08 | 0,06 | 0,08 | - | - |
| Angry | - | 0,13 | 0,83 | 0,04 | - | - | - |
| Neutral | - | - | - | 1,00 | - | - | - |
| Surprised | 0,08 | - | - | 0,10 | 0,78 | - | 0,04 |
| Fear | - | - | 0,08 | - | 0,15 | 0,49 | 0,28 |
| Disgust | - | - | 0,09 | 0,07 | 0,12 | 0,20 | 0,52 |

Figure 7. The Results of Blind Users Test (a) Success Rate of each Expression (b) Confusion Matrix of Facial Expression Recognition

In Figure 7a, the subjects had relatively the same success rate between tests. However, there were significant differences in the recognition of expressions of fear and disgust. After a question and answer session was held on the subject, it was explained that the subject was not good at recognizing information by voice. However, the subjects felt more comfortable after data collection on the second compressor. Judging from Figure 7b, the system test with blind person subject has a total prediction accuracy of 77.06%.

The expression of fear becomes the facial expression that has the biggest mistake. On the expression of disgust and fear, the error of the answer was always on the expression of anger. This is presumably because expressions of anger and disgust have similar facial features or the dataset limitations on expressions of disgust and fear. Other expressions have errors that fall within tolerance limits.

## V. Discussions

In summary, we demonstrate that cloud adoption for facial expression recognition is promising. If it is compared to full on-premises deep learning model training, this approach might be not as practical as we thought before, since we have been modifying everything as suitable as our needs. This problem could be solved by using AI-model services that are

provided by AWS, using Amazon Rekognition (AI service that focused on video and image analysis). Another reason for this is based on our experience, when we handled some problems and errors, we spent quite a large amount of money just for training a model and implementing it to AWS DeepLens device. This means that deep learning models creation must be done by on-premises success, rather by making it online and trained it several times until the optimal tuned model reached.

For future research and study, it is recommended that we use on-premises trained deep learning models and upload it to the cloud storage system. From there, we could create our own Lambda functions and triggers every implementation with cost efficiency and practical usage, rather than understanding every needed service on cloud computing platform.

## VI. Conclusion

From the results of comprehensive research and analysis related to systems and tools, this approach could get an 76.16% accuracy. we learn that cloud computing, especially AWS, need some experience into its development environment due to every service callbacks and triggers do not apply the same function callings. Another point that we can draw is the attention when choosing cloud services. This may be as simple as it sounds, but this is crucial, because to minimalize migration errors, we need to stick to the pre-customized settings by the cloud vendor we choose. After all we have been through during this study, we can conclude that this study can be developed, but cloud computing mastery is necessary for similar type of study.

## References


[1] N. Buduma and N. Locasio, *Fundamentals of Deep Learning*, Boston : O'Reilly, 2017, pp. 11-21.

[2] H. P. Buimer et al., "Conveying facial expressions to blind and visually impaired persons through a wearable vibrotactile device," *PLoS One*, vol. 13, no. 3, pp. 1-16, 2018.

[3] H. Kusuma, M. Attamimi and H. Fahrudin, "Deep learning based facial expressions recognition system for assisting visually impaired persons," *Bulletin of Electrical Engineering and Informatics*, vol. 9, No. 3, pp. 1208–1219, 2020.

[4] Y. Jin, J. Kim, B. Kim, R. Mallipeddi, and M. Lee, "Smart Cane: Face Recognition System for Blind", *Conference on Human-Agent Interaction*, Daegu, 2015, pp. 145-148.

[5] Kaggle. (2013). *Challenges in Representation Learning: Facial Expression Recognition Challenge* [Online]. Available: https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/.

[6] E. Barsoum, C. Zhang, C. C. Ferrer, and Z. Zhang, "Training Deep Networks for Facial Expression Recognition with Crowd-Sourced Label Distribution," *18th ACM International Conference*, 2016.

[7] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *International Conference on Learning Representations*, 2015, pp. 1-14.

[8] K. Pasupa, W. Sunhem, and C. K. Loo, "A Hybrid Approach to Building Face Shape Classifier for Hairstyle Recommender System" *Expert Syst. Appl. 120*, 2019, pp. 14-32.

[9] Z. Yang, T. Dan and Y. Yang, "Multi-Temporal Remote Sensing Image Registration Using Deep Convolutional Features," in IEEE Access, vol. 6, pp. 38544-38555, 2018.

*[10]* AWS Developer Team. (2019). *Amazon SageMaker Developer Guide* [Online]. Available: https://docs.aws.amazon.com/sagemaker