Evaluating the Accuracy of Public Cloud Vendor Face Detection API's

Ashling Malone and John Burns
TU Dublin, Tallaght Campus, Dublin 24, Ireland
Email: ashlingmalone@hotmail.com, john.burns@tudublin.ie

Abstract—**The ability to process human face information is crucial in many areas of government, business, and social media. Facial recognition provides businesses with the ability to provide services that include security, robotics, analysis, human resources, mobile applications, and user interfaces. Users can access their accounts and sign off transactions online just by taking a 'selfie'. Machine Learning algorithms have been developed for face detection in media such as picture images. To recognise a face, the camera software must first detect it and identify the features before making an identification. Face detection is the first step of face recognition. In this research, the face detection APIs from five of the top public cloud vendors of facial recognition software have been tested and evaluated to establish which vendor performs the best for accuracy and to find any significant differences between the vendor APIs. The attributes tested were 'Gender' and 'Age'. Surprisingly, the vendor Amazon Rekognition, IBM and FaceX only offered the attribute age as a range value rather than committing to an exact age. This immediately diminishes the accuracy of their respective APIs. The research proves the weaknesses in API accuracy by testing the resilience of the vendor APIs against degraded images. Azure was the overall winner with Rekognition in second place, Kairos in third, fourth place was IBM and FaceX took last place.**

Index Terms—**algorithms, API, dataset, face detection, face recognition**

I. Introduction

Facial recognition is the preferred method of biometrics as it is easy to deploy and there is no physical interaction required by the end user plus it is extremely fast. It is used to identify and authenticate a person using a set of recognizable verifiable data unique to that person. It does this in two keyways. A 2D or 3D sensor captures a face. It then transforms it into digital data by applying an algorithm, before comparing the image captured to those held in a database. The automated systems can be used to identify or check the identity of individuals in just a few seconds based on their facial features: spacing of the eyes, bridge of the nose, contour of the lips, ears chin etc. [1]. This research focuses on face detection which is the process of detecting and locating human faces in images. It is the first stage of face recognition.

This research is based on a requirement to create an online application for a bank that can offer a secure login assistance to customers who want to make transactions on their savings accounts. The approach was to identify five of the best face recognition public cloud vendors on the market and to test the resilience and accuracy of their Face detection API's. The vendors under observation are Microsoft Azure Face API, Amazon Rekognition, IBM Cloud Visual Recognition, FaceX and Kairos [2]-[7].

There are many limitations in facial recognition for example lighting, posing angles, quality of the images, real time tools verses static image tools. Web services are making it easier to develop applications and cloud technology is making it cheaper and simpler to store and access data. These techniques can be measured by comparing their accuracy with a chosen dataset. This was conducted by manipulating and degrading six images from a carefully predefined dataset and simulating the conditions that can be experienced when taking photos on a mobile phone. The degraded images were then tested for resilience using the vendor API.

The properties used to degrade the images were Blur, Brightness, Contrast, Rotation, Noise, Person not looking at the camera, and Transparency. The data resulting from the tests was analyzed and evaluated to determine if there were any significant differences between the API's and decipher which of the show cased vendors offer the best quality and accuracy in the industry of facial recognition software. This would determine which vendor software if any were suitable for the banking application. [8]-[12].

II. The Experiment

The focus of the experiment was on measuring and testing the performance of face detection API software using the five chosen vendors that were selected for this research paper.

A. The Dataset and Image Manipulating/Degradation

For the purpose of authenticity, a carefully selected group of images was chosen from a dataset called 'VGGFace2' [13]. All images that were downloaded came with the gender, YOB and ethnicity of each person. This was crucial information as the actual measurement of performance was based on the outputs from the attributes; gender and age of the person in the image file being degraded. These images included a combination of

Manuscript received July 25, 2020; revised December 28, 2020.

doi: 10.18178/joig.9.1.20-26

gender, ethnicity, and age as well as males with facial hair, glasses, not looking at camera with black and white images as well as colour. Six images were finally selected from the dataset and were manipulated for testing purposes.

The images were degraded by using filters and manipulation software 'Gimp' and 'Ultra Office Suite'. These were used to mimic conditions that could be faced by a user taking a selfie with a camera. The conditions included blurring, brightness, contrast, not looking at camera, noise, rotation, and Transparency.

The conditions were measured starting at zero degradation with a clean image and degrading the image in 5% intervals until the image was degraded to 100%. These were small enough increments to capture precise results.

It was decided to measure the performance of the API's based on the accuracy of the API's. Accuracy became the variable on the Y-axis as it was the dependent variable. This value would range from 0% to 100%.

The constant variable which was used to measure against the accuracy was Degradation on the X-axis. This value would range from 0 to 100%. The dataset consisted of 117 images on completion of degradation.

B. Testing the Vendor API Software

The next step was to test the API Vendor software using the degraded images for test conditions; Blurring, Brightness, Contrast, Not Looking at Camera, Noise, Rotation and Transparency. The five vendors selected were Microsoft Azure, Amazon Rekognition, FaceX, IBM Bluemix and Kairos. The two attributes that were chosen to be measured were 'gender' and 'age'. The reason these two attributes were chosen was because they were the common denominator attributes provided by all five Vendor API's. The testing was carried out comprehensively on all five vendors with Microsoft Azure being demonstrated in detail below.

C. The Microsoft Azure Face API - Face Detection

Microsoft Azure Face API was the most informative, well documented, and user-friendly option of the five chosen vendors. It provides documentation with running commentary through each stage of automating their API's [2].

Prerequisites:

- Choose a language to use for automation (JavaScript)
- A URL address to access the image being analysed.
- A free subscription-key
- A location address provided on site.
- Code Studio Editor

Method:

Detect faces in an image using REST API and JavaScript

- Initialize the html file by creating a file in Code Editor and save it as a HTML file. Copy the title html code inside the body element of the document from the Azure site to add a basic user interface and a URL field, an Analyze Face button, a response pane, and an image display pane.
- Write the JavaScript. The site provides code to copy into the editor. It sets up the JavaScript code that calls the Face API. This section includes adding the subscription key which offers you access to the API code. The free subscription key is generated in the region chosen as the location.
- When finished entering the code the parameters and Post code is included.
- The html file is saved to the stored location.
- To run the script, the user opens the html file in the browser, then clicks the Analyze Face button, the application displays the image from the given URL and prints out a JSON string of face data.

Fig. 1 shows image type used for the experiment; female of Middle Eastern origin born in 1974. In the second example, Fig. 2 shows image degraded by increasing the contrast by 75%. The over exposure level is visible in the image.

Fig. 3 is an example of the output from Azure FACE api after running the code and opening the html file in the browser.

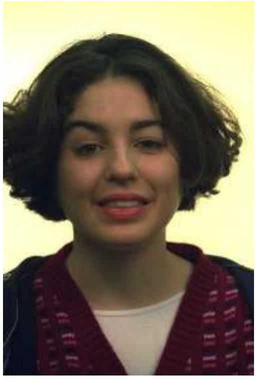


Figure 1. 0% contrast.

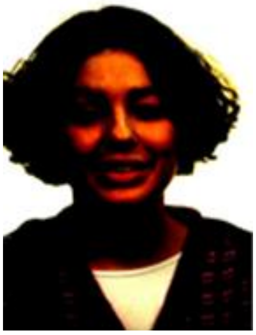
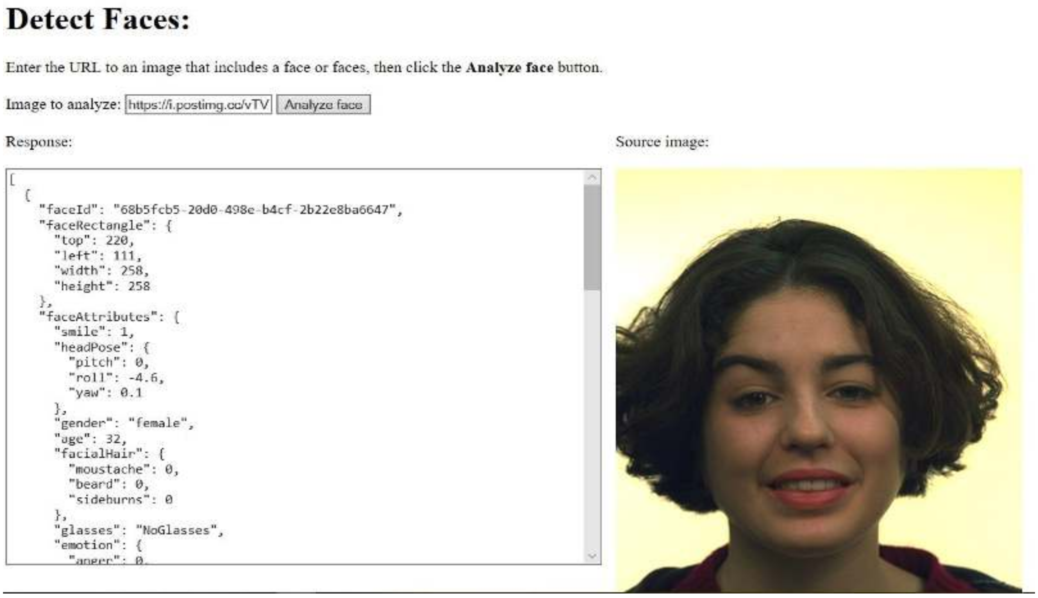


Figure 2. 75% contrast.

D. Creating an Algorithm for Scoring Age

For this experiment, an algorithm was created in Excel for vendors; FaceX, IBM and Rekognition to penalise their final API score because one of the attributes which was age was offered as a range rather than committing to an exact number. Fig. 4 shows the steps of the algorithm.



Detect Faces:

Enter the URL to an image that includes a face or faces, then click the **Analyze face** button.

Image to analyze: https://i.postimg.cc/vTV [Analyze face]

Response:

```
[
  {
    "faceId": "68b5fcb5-20d0-498e-b4cf-2b22e8ba6647",
    "faceRectangle": {
      "top": 220,
      "left": 111,
      "width": 258,
      "height": 258
    },
    "faceAttributes": {
      "smile": 1,
      "headPose": {
        "pitch": 0,
        "roll": -4.6,
        "yaw": 0.1
      },
      "gender": "female",
      "age": 32,
      "facialHair": {
        "moustache": 0,
        "beard": 0,
        "sideburns": 0
      },
      "glasses": "NoGlasses",
      "emotion": {
        "anger": 0,
```

Figure 3. With Azure the following characteristics are returned in the results.

Algorithm Steps

Step 1

Input age range in cell A and B

Step 2

(A2:B2)/2

Step 3

Input age of model in cell D

Step 4

SUB(D2, C2)

Step 5

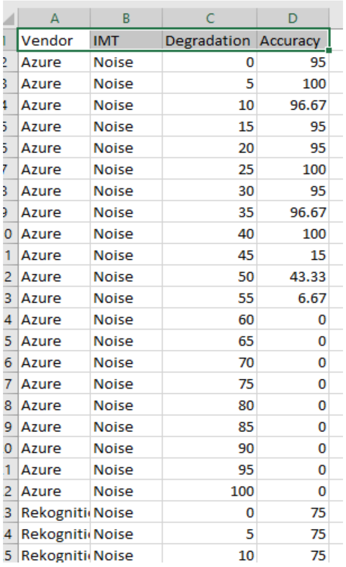
(C2/D2*100)

Step 6

100 – F2/2 = Total to be deducted from score

Figure 4. Steps of the algorithm in excel.

E. Results of Experiment



Vendor	IMT	Degradation	Accuracy
Azure	Noise	0	95
Azure	Noise	5	100
Azure	Noise	10	96.67
Azure	Noise	15	95
Azure	Noise	20	95
Azure	Noise	25	100
Azure	Noise	30	95
Azure	Noise	35	96.67
Azure	Noise	40	100
Azure	Noise	45	15
Azure	Noise	50	43.33
Azure	Noise	55	6.67
Azure	Noise	60	0
Azure	Noise	65	0
Azure	Noise	70	0
Azure	Noise	75	0
Azure	Noise	80	0
Azure	Noise	85	0
Azure	Noise	90	0
Azure	Noise	95	0
Azure	Noise	100	0
Rekogniti	Noise	0	75
Rekogniti	Noise	5	75
Rekogniti	Noise	10	75

Figure 5. Example of CSV file content.

On completion of gathering the results for all five vendors, a CSV file was created with the entire results as a table. This consisted of 636 data entries with percentage results. The file consisted of four headings; Vendor, IMT (Image Manipulation Type), Degradation and Accuracy and was subdivided into seven separate files. Each of the seven files was grouped by degradation type which were Blur, Brightness, Contrast, Noise, Not looking at Camera, Rotation and Transparency. Fig. 5 shows a section of this CSV file.

III. Analysis of Results - Brightness Category

The next stage of the research was to set up the results in r studio and run statistical tests to analyse the accuracy and performance of the API results. This was conducted for each category using the respective CSV results files. In this chapter a subset of the testing is described using ‘Brightness’ as the example.

A. Scatterplots: Observations of the Data

For the first stage of testing, scatterplots were created in r for each vendor. The purpose of using the scatterplot was to analyze how the X and Y-variables relate to each other. Scatterplots are the best method for showing a non-linear pattern and the range of data flow, i.e., the maximum and minimum value, can be easily determined. A Linear model is the right model to test when there are two variables - X and Y which are continuous.

Fig. 6 is an example of the scatterplots; the X-axis represents the degradation measuring from 0% to 100%. The Y-axis represents the accuracy of the API measuring from 0% to 100%. As can be seen in all five scatterplots, there are phase changes or jumps in the data. For the experiment, the images were degraded by gradually brightening them at intervals of 5%, recording the API output each time until they were either degraded up to 100% or up to the point where the API failed. With all five scatter plots the API quality goes from high quality readings of 80% to 90% accuracy on the Y-axis to breaking completely at 70% to 80% degradation on the X-axis for all vendors. This is demonstrated by the slope on each graph showing the negative relationship between the X and Y variables. The pattern for all five vendors is reasonably similar.

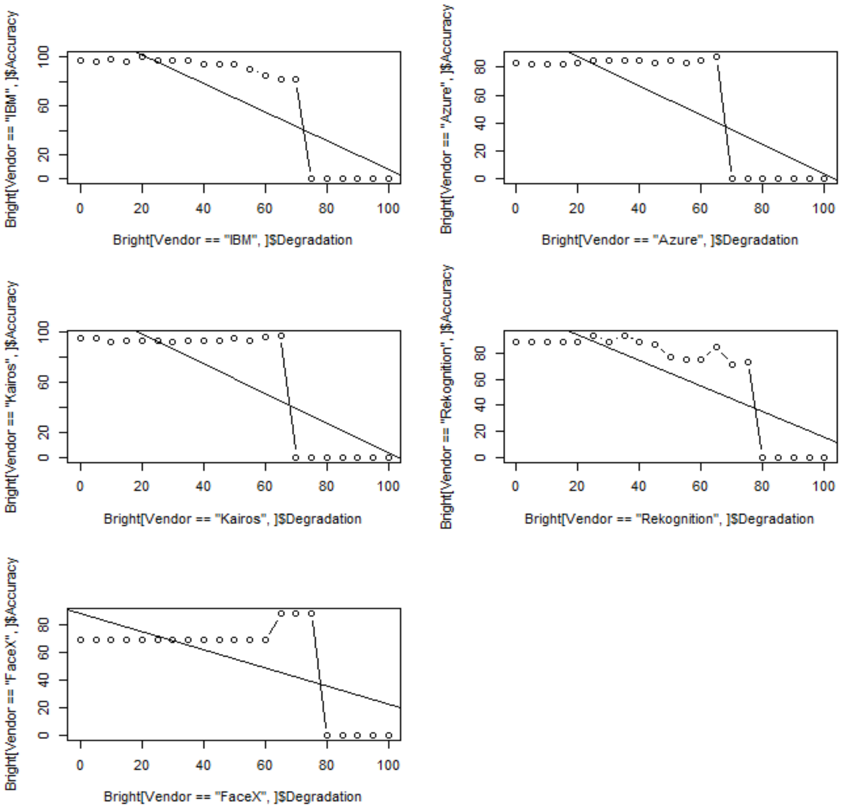


Figure 6. Brightness scatterplots for the five vendors.

B. Shapiro Wilk Test

For the second stage of testing, a Shapiro-Wilk test was performed in r on each vendor. The Shapiro Wilk test is a one tailed test that tests for normal distribution. The test gives a *p*-value; small values indicate the sample is not normally distributed. There are two hypotheses in a Shapiro Wilk test, the null hypothesis (HO) that the residuals are normally distributed. If the *p*-value is greater than the chosen alpha level, then the null hypothesis cannot be rejected as there is no evidence that the residuals tested are not normally distributed. The alternative hypothesis (H1) that if the *p*-value is less than the chosen alpha level, then the null hypothesis is rejected as there is evidence that the residuals tested are not normally distributed.

The Table I below shows the result of the Shapiro-Wilk test for each vendor for the category Brightness. The *p*-value for each of the vendors is greater than the chosen alpha level of 0.05, so we cannot reject the null hypothesis that the residuals are normally distributed.

TABLE I. SHAPIRO RESULTS TABLE FOR BRIGHTNESS

Vendor	*p*-value	Result
IBM	0.2144	The *p*-value is > 0.05
Azure	0.3439	The *p*-value is > 0.05
Kairos	0.3256	The *p*-value is > 0.05
Rekognition	0.2408	The *p*-value is > 0.05
FaceX	0.258	The *p*-value is > 0.05

C. Analysis of Variance – ANOVA

For the third stage of testing, the mean of each vendor was taken for Brightness as seen in Table II below. Then an Anova test was performed on all means. An Anova test was chosen for this experiment because it allows comparisons between the means of three or more groups of data, in this case there are five groups of data. It is a hypothesis-based test, meaning that it aims to evaluate multiple mutually exclusive theories about the data.

TABLE II. MEAN OF FIVE VENDORS FOR BRIGHTNESS

Vendor	Azure	FaceX	IBM	Kairos	Rekognition
Mean	55.92238	55.49381	66.75857	62.75952	64.71190

There are two possible hypotheses in a one-way Anova, the null hypothesis (HO) that there is no difference between the groups and equally between the means. The alternative hypothesis (H1) that there is a difference between the means and groups, that at least one sample mean is not the same as the others.

TABLE III. P-VALUE RESULT FROM ANOVA

P–value (PR(>F))
0.847

The value in Table III shows the result of running the Anova test in r. If the p-value is greater than alpha; the difference between the means is not statistically significant. If the p-value is greater than the significant level, there is not enough evidence to reject the null hypothesis that the population means are all equal.

In this case the p-value is greater than Alpha, which is 0.05, therefore there is no difference between the means of the five vendors. This is reinforced by viewing the means of the five vendors in Table II.

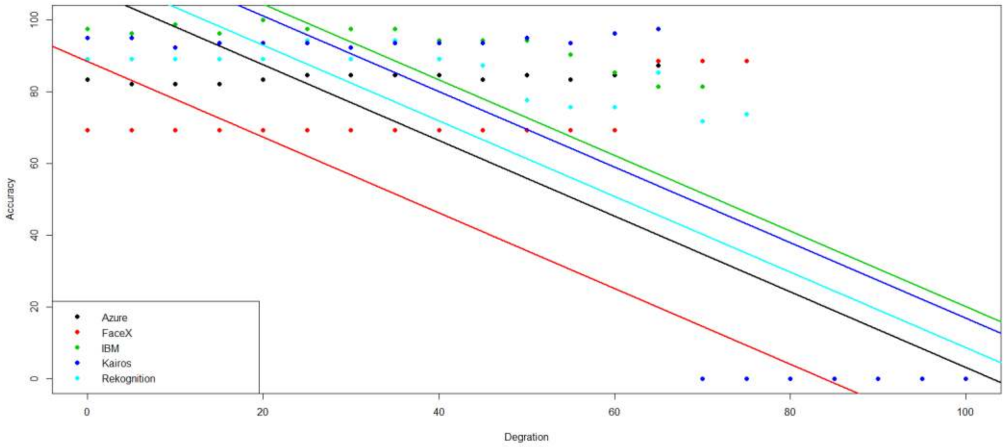


Figure 7. ANCOVA model with slopes for five vendors – Brightness.

D. Analysis of Covariance – ANCOVA

For the fourth stage of testing, an Ancova test was performed in r on all vendors for category Brightness**.** Ancova which is the analysis of covariance evaluates whether the means of the dependent variable are equal across levels of a categorical independent variable [13]. It is a combination of an Ancova and a regression analysis. In basic terms, the Ancova examines the influence of an independent variable or a dependent variable while removing the effect of the covariate factor. Ancova is being used because the five regression lines can be compared to each other; The Ancova tells us whether the regression lines are different to each other in either slope or intercept. An Ancova is graphed with a scatterplot, the independent variable is on the X axis and the dependent variable is on the Y axis. Each vendor is represented by a different colour with a legend identifying the vendors on the graph as shown in Fig. 7 above.

This shows the results after running the Ancova test commands in r. From viewing the graph, the correlation between the x and y variables is negative because as one variable increases, the other variable decreases**.** The slope on the x -axis increases as the y variable accuracy decreases. As the degradation of the image increases, the accuracy of the API's decrease and the APIs stop working for each vendor. This generally happens at about 70% to 80% degradation.

Table IV lists the Vendor API performance which was established from taking the readings from the Ancova graph. IBM, which is the top slope and coloured green is the winner as it has the highest Accuracy while FaceX which is coloured red performed the worst with the least accuracy.

TABLE IV. CATEGORY WINNER RESULTS TABLE FOR BRIGHTNESS

Vendor	Position
IBM	**1st Place**
Kairos	**2nd Place**
Rekognition	**3rd Place**
Azure	**4th Place**
FaceX	**5th Place**

IV. CONCLUSION

On completion of the research, there were 117 images in the dataset degraded for testing the APIs of the five vendors. 636 data results were recorded in response to the testing and that data was statistically evaluated in r studio to establish the winning API vendor and the differences between the accuracy of the API's.

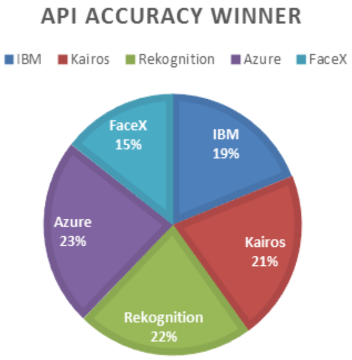


Figure 8. Overall API accuracy winners.

Fig. 8 shows the winner for overall API Accuracy with Microsoft Azure taking first place, Rekognition second place, Kairos in third, fourth place was IBM with FaceX taking last place. Overall, looking at the statistics there was not a huge difference in performance between the vendors.

Surprisingly, the vendor Amazon Rekognition only offered the attribute ‘age’ as a range value for its face detection API. This limits the use of this API and one would question how accurate the API could possibly be with a broad age range value. IBM and FaceX API also fail to commit to an exact age. FaceX had an interval of twenty years for their age range which is huge when you consider that it could be the difference between detecting a person’s age as being in their early twenties when they are actually in their forties.

Recommendations for future work would be to create and offer algorithms that can call API’s with exact attributes and not ranges to ensure market confidence in the accuracy on the API.

A conclusion was drawn from reading the Azure website, that Microsoft makes it clear that their API is very dependent on outside influences such as lighting, angles, image size and oscillations. The API is only as good as the image it is working with. Arguably, the API algorithm should be robust and intelligent enough to recognise obstructions and retain accuracy during performance. Face detection technology can detect frontal or near-frontal faces in a photo, regardless of orientation, lighting conditions or skin colour [14].

From analysing the results of the degraded APIs, there was a negative relationship between the variables in most of the groups or categories of data with Rotation being the exception. This shows the weaknesses in the API’s, starting with a high performance mostly breaking completely at a certain threshold.

Most scatterplots had phase changes, jumps in the data or changes in the way the API performed often going from high performance to breaking completely instantly. This demonstrated how the APIs responded to the degradation of the images.

It is clear from the experiment that improvements can be made to all vendor API’s in all areas that were tested, particularly for FaceX which scored the lowest overall.

The testing could be carried out more profoundly with a larger dataset and working in increments smaller than 5% intervals. This would also require more time and effort.

From current state of the art trends, the results back up the current feeling that Face recognition is not a 100% reliable source of biometrics and is prone to error under certain conditions. It is a fascinating, quick, and easy biometric technique and is capturing the interest of every technology company dealing with AI and machine learning. With improvements it has the capacity for further beneficial practices in the future.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

This paper was researched and written by Ashling Malone as a Thesis for a Masters and supervised by Dr. John Burns. Both authors approved the final version.

REFERENCES

[1] R. Safi. (2019). Facial recognition system – The new future of biometrics identification. [Online]. Available: https://apiumhub.com/tech-blog-barcelona/facial-recognition-biometrics-identification/

[2] P. Farley, *et al*. (2019). Microsoft Azure. Quickstart: Detect faces in an image using the Face REST API and Python. [Online]. Available: https://docs.microsoft.com/en-us/azure/cognitive-services/Face/QuickStarts/Python

[3] Amazon Web Services. (2019). Face detection. [Online]. Available: https://eu-west-1.console.aws.amazon.com/rekognition/home?region=eu-west-1#/face-detection

[4] FaceX. (2019). Face detection. [Online]. Available: https://facex.io/

[5] IBM. (2019). Detect faces in an image. [Online]. Available: https://cloud.ibm.com/docs/services/visual-recognition?topic=visual-recognition-getting-started-tutorial

[6] Kairos. (2019). Face detection. [Online]. Available: https://www.kairos.com/demos

[7] S. G. Jung, J. An, H. Kwak, J. Salminen, and B. J. Jansen, “Assessing the accuracy of four popular face recognition tools for inferring gender, age, and race,” in *Proc. International AAAI Conference on Web and Social Media*, 2018, pp. 624-627.

[8] S. T. Graham and X. Liu, “Critical evaluation on jClouds and cloudify abstract APIs against EC2, Azure and HP-Cloud,” presented at IEEE 38th International Computer Software and Applications Conference Workshops, Vasteras, Sweden, July 21-25, 2014.

[9] B. Yang, J. Yan, Z. Lei, and S. Z. Li, “Fine-Grained evaluation on face detection in the wild,” presented at 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition, Ljubljana, Slovenia, May 4-8, 2015.

[10] M. Phankokkruad and P. Jaturawat, “An evaluation of technical study and performance for real-time face detection using web real-time communication,” presented at International Conference on Computer, Communications, and Control Technology, Langkawi Island, Kedah, Malaysia, September 2-4, 2014.

[11] K. Dang and S. Sharma, “Review and comparison of face detection algorithms,” presented at 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence, Noida, India, Jan. 12-13, 2017.

[12] Q. Cao, L. Shen, W. Xie, O. Parkhi, and A. Zisserman, “VGGFace2: A dataset for recognising faces across pose and age,” in *Proc. 13th IEEE International Conference on Automatic Face & Gesture Recognition*, 2018, pp. 67-74.

[13] S. S. Mangiafico, *Regressions, Analysis of Covariance. An R Companion for the Handbook of Biological Statistics*, version 1.3.2., 2015.

[14] SightCorp. (2019). Face detection. [Online]. Available: https://sightcorp.com/face-detection

Ashling Malone was born in Dublin Ireland. Initially Ashling studied Mechanical Engineering and worked in the Electronics industry for several years. She then diverted into computing, studied Computing, and worked as a Consultant in Test Management, Business Analysis, Systems Analysis and Project Management. She completed an M.Sc. in Applied IT Architecture at the Technological University Dublin, Tallaght Campus in 2019.

John Burns was born in Dublin, Ireland and is a senior lecturer in Computing at the Technological University Dublin, Tallaght Campus. He is Interested in Distributed Computing. John has many publications including ‘Implementation of a scalable real time canny edge detector on programmable SOC’, 2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA) (IEEE). “CUDA-enabled Optimisation of Technical Analysis Parameters”, Proceedings of the 2012 IEEE/ACM 16th International Symposium on Distributed Simulation and Real Time Applications.