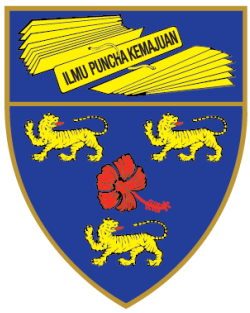


UNIVERSITI MALAYA

Soft Computing (WIX3001)

Semester 2 2022/2023

Assignment 2

Report

Lecturer : Dr Liew Wei Shiung

Group Members :

Name	Matric No.
Janice Chong See Wai	S2132420
Jasmine Chong See Yan	S2132419

Table of Content

1.0 Problem Identification

In this report, we will be buliding a model to identify writers based on handwriting using Python in Jupyter Notebook.

2.0 Data Collection and Preprocessing

2.1 Data Collection

Number of classes: 2

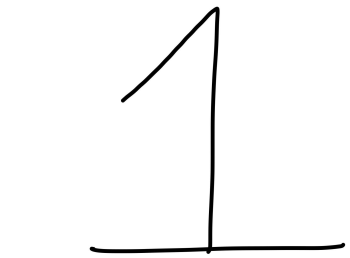
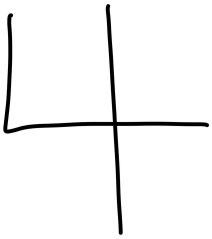
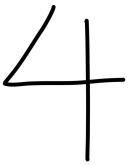
Our group comprises only 2 members. Thus, we will only have 2 classes as the target, namely, Janice and Jasmine.

Total number of data: 900 samples

Our datasets are created by writing digitally, using an iPad, with a total of 450 samples per person which include:

- 200 digits (0-9)
- 250 alphabets (a-z except ‘l’ and ‘o’ due to their similarities with digits 1 and 0)

Examples of data:

Janice	Jasmine
	
	

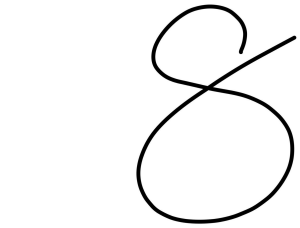
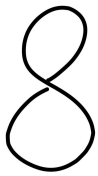
	

Table 1: Samples of numeric data

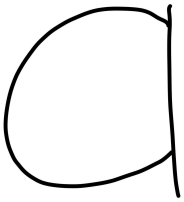
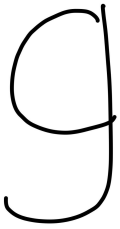
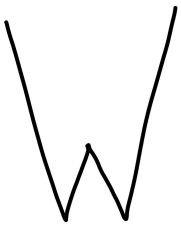
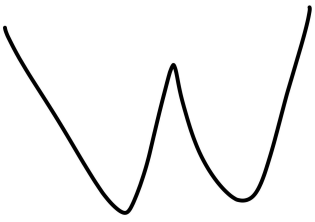
Janice	Jasmine
	
	
	

Table 2: Samples of alphabetic data

2.2 Data Preprocessing

2.2.1 Resize Image

***refer Resize-Image.ipynb*

Initial data format:

- JPG format
- 2182 x 1538 pixels
- RGB colour mode

Using the Python Image Library (PIL), we will process the data to change the data format into something similar to the MNIST format. We take the following steps:

1. Load the image using the Image.open() function from PIL.
2. Resize the image using the thumbnail method:
    - Able to maintain the aspect ratio when resizing
    - Takes in 2 arguments; target size and Image.ANTIALIAS (to ensure high-quality downscaling)
3. Create a new image using the Image.new() function with 3 arguments:
    - Target size as the canvas size (28 x 28 pixels)
    - 'L' to convert the image to greyscale
    - '255' to set the background colour to white
4. Paste the resized image onto the blank canvas using:
    - paste() function of the resized image
    - Offset is calculated by finding the difference between the target size and the original image size to determine the amount of padding required on each side (horizontally and vertically) of the resized image so that the image will be pasted at the center of the canvas
    - The calculated offset represents the number of pixels that the resized image needs to be shifted horizontally and vertically to be centered within the canvas of the target size

Final data format:

- JPG format
- 28 x 28 pixels
- L colour mode (greyscale)

2.2.2 Data Partitioning

***refer Writer-Identification-v7.ipynb*

We split our datasets into training and testing sets using the train_test_split() from the sklearn library. The test set size is set to 20% and the remaining 80% is the training set and random state of 42 to ensure that training and testing sets are the same every time we run it for consistency.

2.2.3 Normalisation

***refer Writer-Identification-v7.ipynb*

We perform normalisation by dividing our data with the pixel value of 255 to ensure that data features are all on a similar scale of 0 to 1. We then convert our target labels into one-hot encoding representations using the to_categorical() function.

3.0 Soft Computing Method

***refer Writer-Identification-v7.ipynb*

We used Convolutional Neural Networks (CNN) to build our model.

3.1 Training the Model

The RandomSearch tuner is used to obtain the best parameters that result in the highest accuracy of the model with the following criteria:

- The number of filters in the convolutional layer is set using a search space hp.Int() which allows us to tune our hyperparameters;
    - The number of filters of the convolutional layer is set as; 16 as a minimum, 128 as the maximum
    - Step of 16
- The number of dense units in the hidden dense layer is also set using the search space hp.Int();
    - The number of dense units of the hidden dense layer is set as; 64 as a minimum, 256 as the maximum
    - Step of 64
- The activation function used in both layers is the Rectified Linear Unit (ReLU).
- The output dense layer is initialised with 2 (number of class) dense units and softmax as the activation function.
- Tuner will perform the search for a maximum of 10 trials with 10 epochs.
- Parameters which result in the highest accuracy will be saved.

To improve the model's accuracy, we have experimented with different parameters combination. The results are recorded in Table 3.

4.0 Results

***refer Writer-Identification-v7.ipynb*

4.1 Experiment Results

Experiment Number	Convolutional Layer		Hidden Dense Layer		Number of Epoch	Best Accuracy
	Number of Layers	Best Number of Filters	Number of Layers	Best Number of Dense Units		
1	1	64	1	128	10	0.9167
2	1	64	2	128	10	0.8000
3	2	64	1	128	10	0.7278
4	1	64	1	128	15	0.9278
5	1	64	1	128	5	0.7611

Table 3: Results of experiments

Based on Table 3, taking Experiment 1 as the control, the increase in the number of layers (Experiment 2 and Experiment 3) and the decrease in the number of epochs (Experiment 5) does not improve the model's accuracy. However, we find that increasing the epoch number (Experiment 4) did in fact improve the model's accuracy. Thus, we can conclude that the best model is obtained from Experiment 4 as it has the highest accuracy of 0.9278 and the loss of 0.3425 with:

- 1 convolutional layer of 64 filters
- 1 hidden dense layer of 128 dense units
- Number of the epoch is 15

5.0 Analyse and Interpret

***refer Writer-Prediction.ipynb*

We prepared an additional set of digits and alphabets (a total of 34 samples per person) as our input data to allow the model to make predictions on who is the writer for those sets of digits and alphabets.

The input images will be resized to 28 x 28 pixels and are calibrated using temperature scaling to increase the confidence value. The reason for us performing the calibration is because initially, our confidence value for the predictions is very low (approx. 0.4 - 0.5) which does not reflect our model's accuracy of 0.9278. Thus, the input images are calibrated to increase confidence when making predictions.

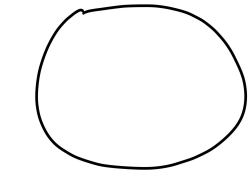
Here we will analyse the model's prediction and classify which digit and alphabet have the highest accuracy and the worst accuracy.

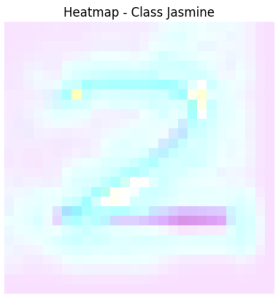
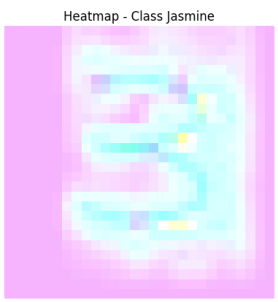
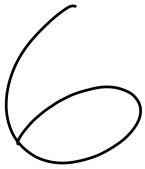
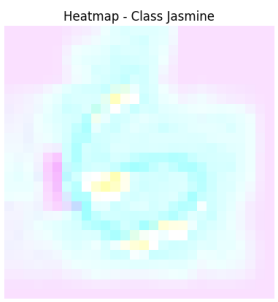
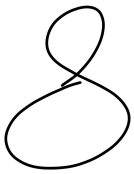
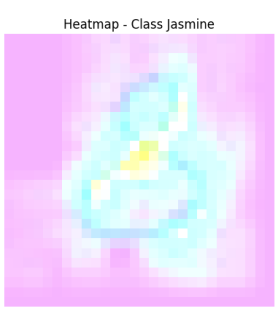
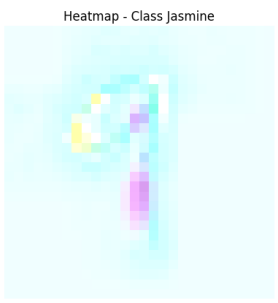
- Highest accuracy = Correct predictions with the highest confidence level
- Worst accuracy = Incorrect prediction but with the highest confidence level

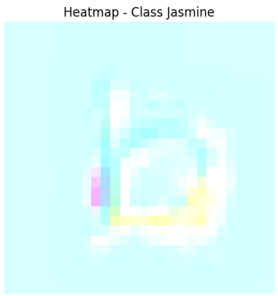
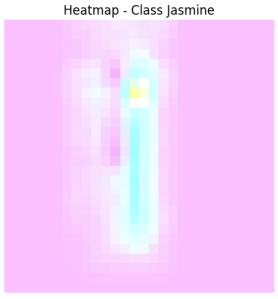
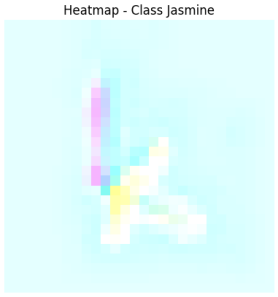
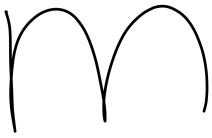
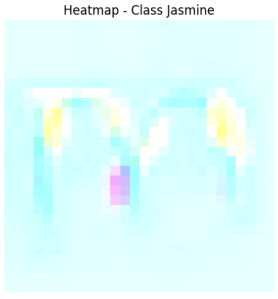
We have also included a visualised interpretation/explainable information of our prediction model in the form of heatmaps obtained using the GradCAM explainer from tf.explain. GradCAM is a technique used for increasing the transparency of Convolutional Neural Network (CNN)-based models by visualising the portions of the input that are "important" for these models' predictions and is also known as visual explanations (Chetoui, M., 2021).

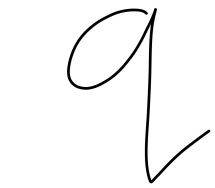
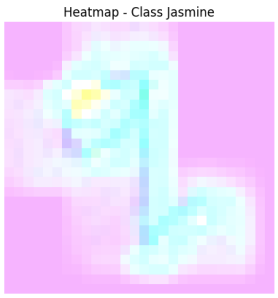
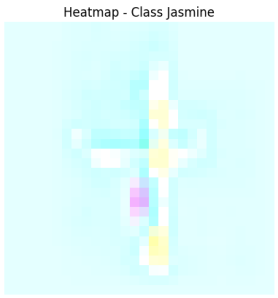
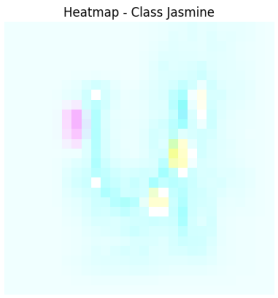
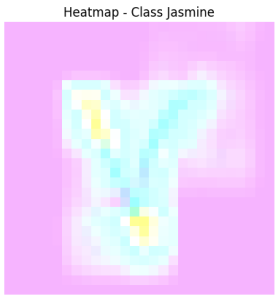
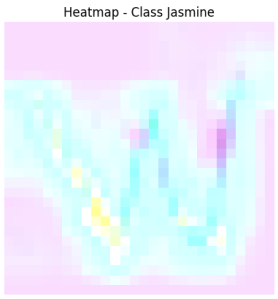
5.1 Jasmine's Actual Handwriting

The first table (refer to Table 4) shows the correct classifications for Jasmine's handwriting while the second table (refer to Table 5) shows the incorrect classifications with their respective confidence values.

Handwriting (Input Image)	Heatmaps	Model Classification	Confidence
		Jasmine	0.9922

		Jasmine	0.6965
		Jasmine	0.9988
		Jasmine	0.9261
		Jasmine	0.9670
		Jasmine	0.9992

		Jasmine	0.9981
		Jasmine	0.9999
		Jasmine	0.8215
		Jasmine	0.9794
		Jasmine	0.9999

		Jasmine	0.9753
		Jasmine	0.9999
		Jasmine	0.9949
		Jasmine	0.7191
		Jasmine	0.9999

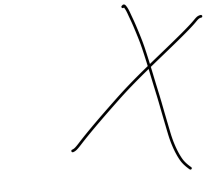
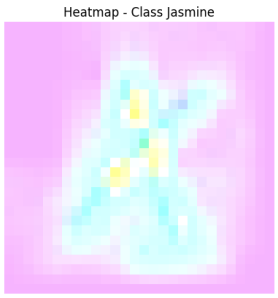
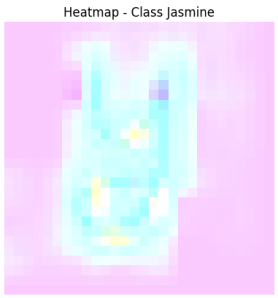
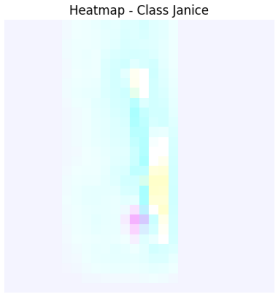
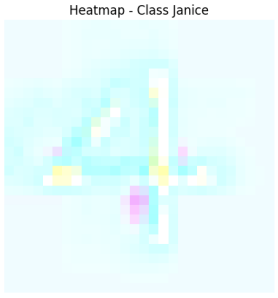
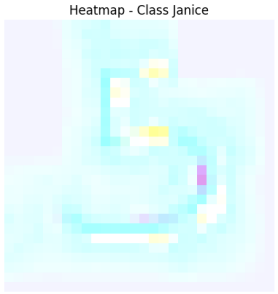
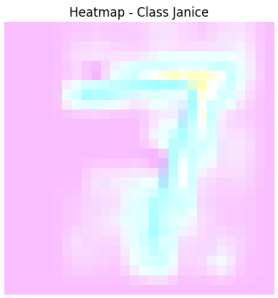
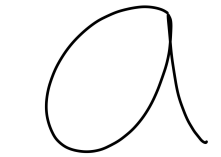
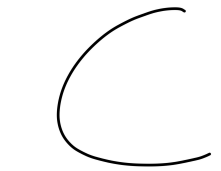
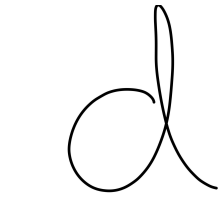
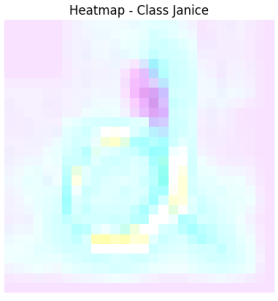
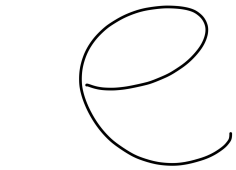
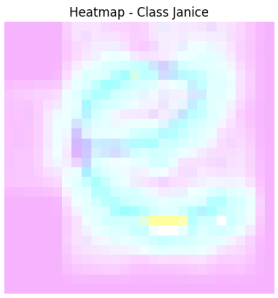
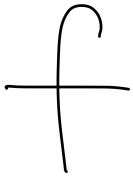
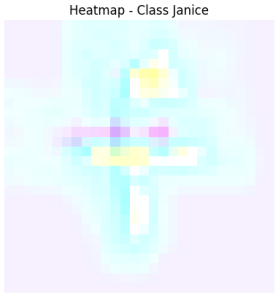
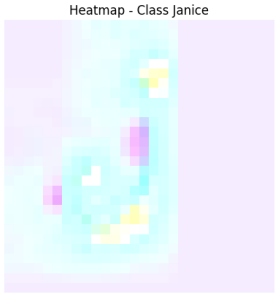
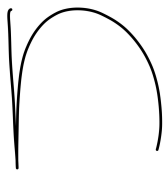
		Jasmine	0.9999
		Jasmine	0.9958

Table 4: Jasmine's correct classifications

Handwriting (Input Image)	Heatmap	Model Classification	Confidence
		Janice	0.5837
		Janice	0.9986

		Janice	0.9997
		Janice	0.9877
		Janice	0.9999
		Janice	0.9999
		Janice	0.9999

		Janice	0.9999
		Janice	0.9992
		Janice	0.9999
		Janice	0.9999
		Janice	0.9999

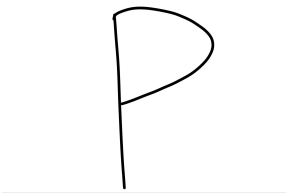
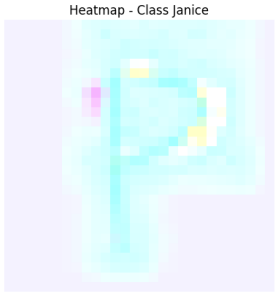
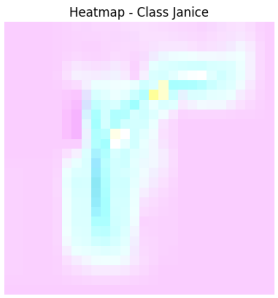
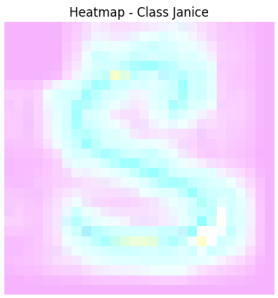
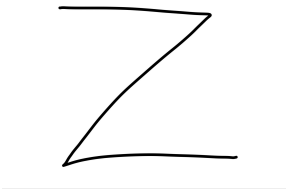
		Janice	0.9880
		Janice	0.9994
		Janice	0.9999
		Janice	0.9999

Table 5: Jasmine's incorrect classifications

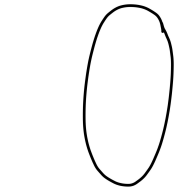
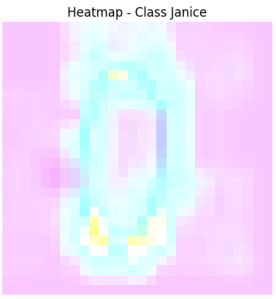
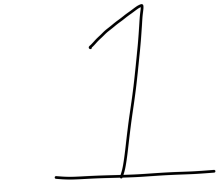
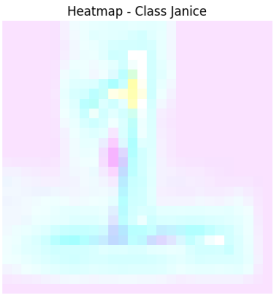
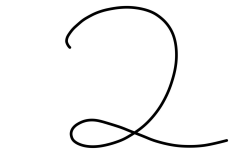
5.1.1 Jasmine's Results

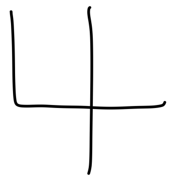
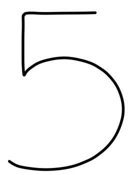
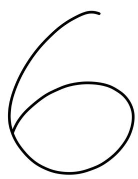
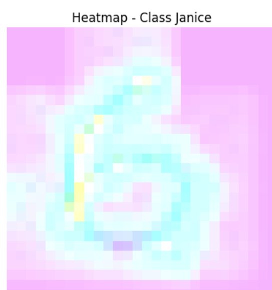
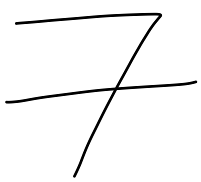
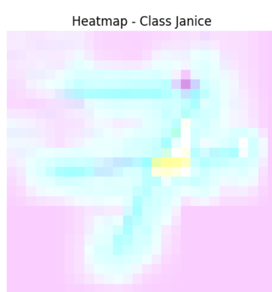
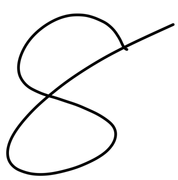
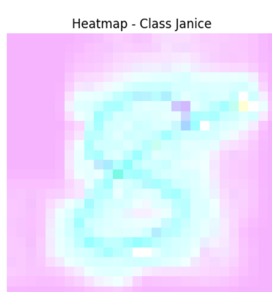
The digit with the best accuracy (0.9992): 9
The alphabet with the best accuracy (0.9999): h, m, t, w, x

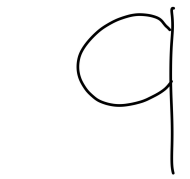
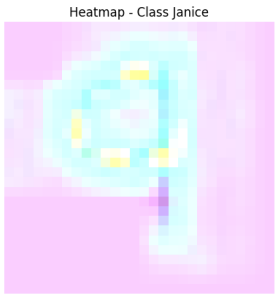
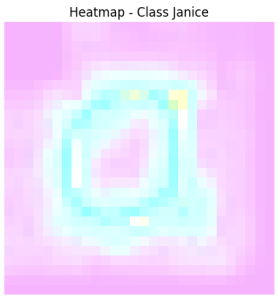
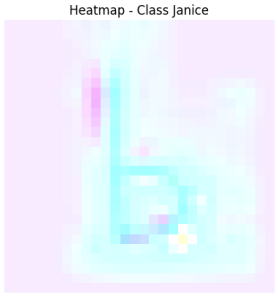
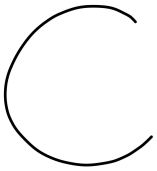
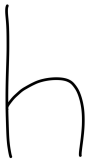
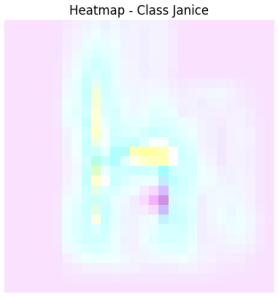
The digit with the worst accuracy (0.9997): 5
The alphabet with the worst accuracy (0.9999): a, c, d, e, g, j, n, s, z

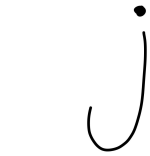
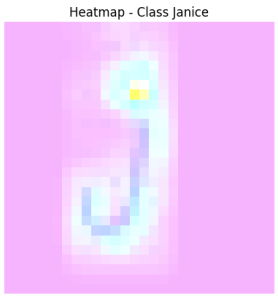
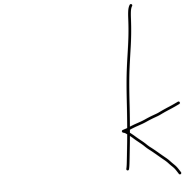
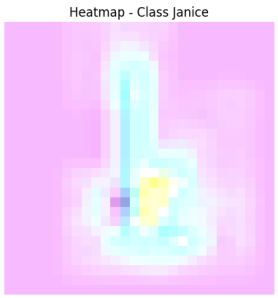
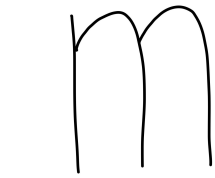
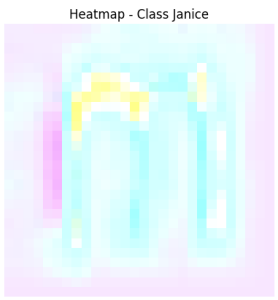
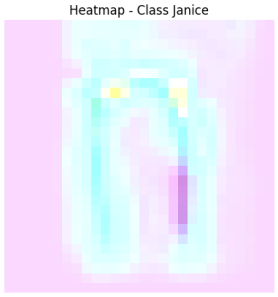
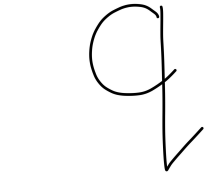
5.2 Janice's Actual Handwriting

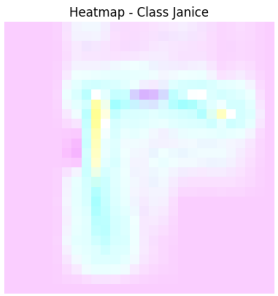
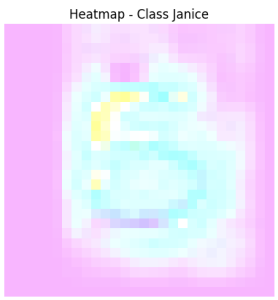
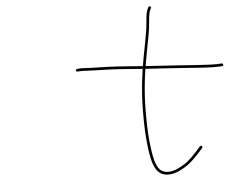
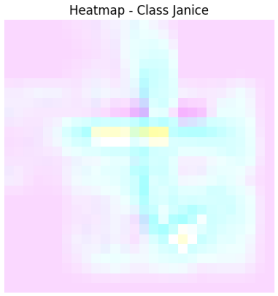
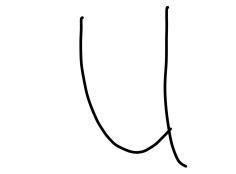
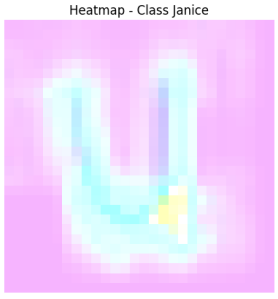
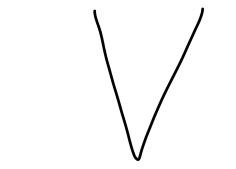
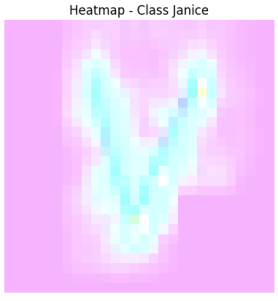
The first table (refer Table 6) shows the correct classifications for Janice's handwriting while the second table (refer Table 7) shows the incorrect classifications with their respective confidence values.

Handwriting (Input Image)	Heatmap	Model Classification	Confidence
		Janice	1.0
		Janice	0.9999
		Janice	0.9999
		Janice	0.9999

		Janice	0.9999
		Janice	0.9999
		Janice	0.9999
		Janice	0.9999
		Janice	0.5862

		Janice	0.9999
		Janice	0.9956
		Janice	0.9522
		Janice	0.9411
		Janice	0.9965

		Janice	0.9996
		Janice	0.9992
		Janice	0.9999
		Janice	0.9996
		Janice	0.5302

		Janice	0.9990
		Janice	0.9060
		Janice	0.9999
		Janice	0.9999
		Janice	0.9970

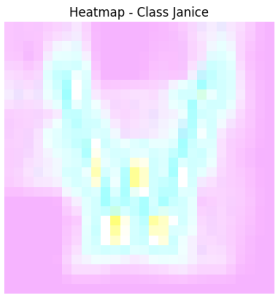
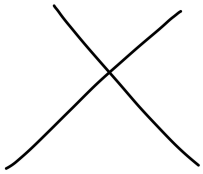
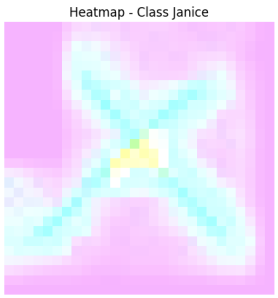
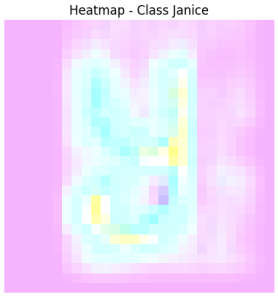
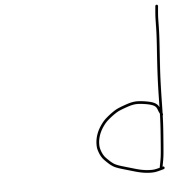
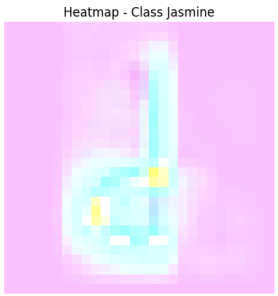
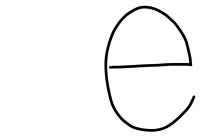
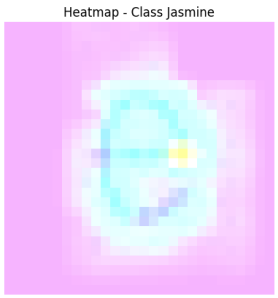
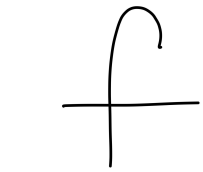
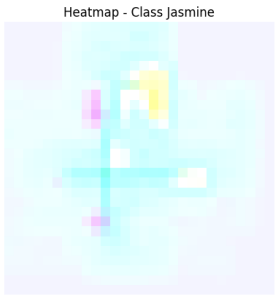
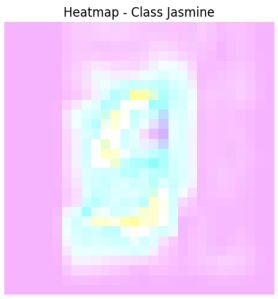
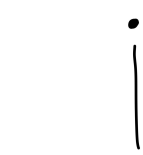
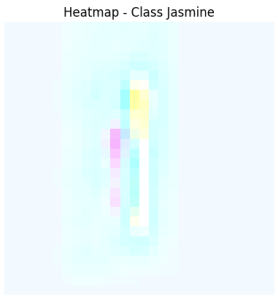
		Janice	1.0
		Janice	0.9999
		Janice	0.9999
		Janice	0.9999

Table 6: Janice's correct classifications

Handwriting (Input Image)	Heatmap	Model Classification	Confidence

		Jasmine	0.9999
		Jasmine	0.9999
		Jasmine	0.8445
		Jasmine	0.9426
		Jasmine	0.9565

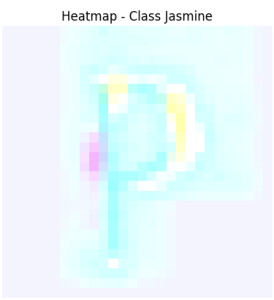
		Jasmine	0.8113

Table 7: Janice's incorrect classifications

5.2.1 Janice's Results

The digit with the best accuracy (1.0): 0
The alphabet with the best accuracy (1.0): w

The digit with the worst accuracy: None. All digits were correctly classified (Although the digit 8 is correctly classified, the confidence level is very low i.e., 0.5862 while the rest have an average of 0.9999 confidence level)
The alphabet with the worst accuracy (0.999): d, e

5.3 Outcomes Summary

Based on the tables above (refer Table 4, Table 5, Table 6 and Table 7), we can conclude that the alphabet d and e have the worst classification as the model failed to classify them correctly for both of the writers' handwriting. Although digits were all classified correctly for Janice's handwriting, we cannot conclude that the model can classify digits better as compared to alphabets. This is because the model failed to classify the digits 1, 4, 5 and 7 correctly in Jasmine's handwriting.

However, we can conclude that Janice's handwriting has the best accuracy as most of the correct handwriting classifications (28/34, 82.4% correct classifications) are Janice's. Whereas, Jasmine's handwriting has the worst accuracy with the majority of her handwriting being wrongly classified (18/34, 52.9% correct classifications only).

7.0 References

Chetoui, M. (2021). Gradient-weighted Class Activation Mapping - Grad-CAM-. *Medium*. https://medium.com/@mohamedchetoui/grad-cam-gradient-weighted-class-activation-mapping-ffd72742243a

Christianversloot. (2022). *machine-learning-articles/neural-network-activation-visualization-with-tf-explain. md at main · christianversloot/machine-learning-articles*. GitHub. https://github.com/christianversloot/machine-learning-articles/blob/main/neural-network-activation-visualization-with-tf-explain.md

Clark, J. (2023). *2.7.0*. Pillow (PIL Fork). https://pillow.readthedocs.io/en/stable/releasenotes/2.7.0.html

Geoff Pleiss. (2017). Calibrating Neural Networks. https://geoffpleiss.com/nn_calibration

Mohd Sanad. (2023). Image Classification Using CNN (Convolutional Neural Networks). *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/

Rosebrock, A. (2021). *Grad-CAM: Visualize class activation maps with Keras, TensorFlow, and Deep Learning - PyImageSearch*. PyImageSearch.

https://pyimagesearch.com/2020/03/09/grad-cam-visualize-class-activation-maps-with-keras-tensorflow-and-deep-learning/

Srivignesh, R. (2021). Tuning Hyperparameters of An Artificial Neural Network Leveraging Keras Tuner. *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2021/06/tuning-hyperparameters-of-an-artificial-neural-network-leveraging-keras-tuner/

Wolfe, C. R., PhD. (2022). Confidence Calibration for Deep Networks: Why and How? Medium. https://towardsdatascience.com/confidence-calibration-for-deep-networks-why-and-how-e2cd4fe4a086