Optimizing Few-Shot Image Classification Berrijam Jam

COMP9417 External Project T1, 2024

DATA MAVERICKS

Gardner Dowling(z5367676)

Vishesh Malik(z5444643)

Devarshi Prakashbhai Patel(z5447832)

Sathwik Bharadwaj Udupi(z5465111)

Negar Mahdavi Goloujeh (z5461835)

1. INTRODUCTION

The Berrijam Jam external task focuses on the current issue in the AI industry of few-shot image classifiers. Normally when training an image classification A.I, vast volumes of training data are expected for each class when the classes differ greatly. However, in the given data sets, not only are the classes for each question visually similar, but there are only 5 training images for each class. The task is to create a highly generalisable AI model that can automatically train on very few and predict to a high degree of accuracy 5 different binary image classification problems, 2 of which are unseen. If we can build highly accurate models that require minimal data for an automated training process, the bar of entry into use is largely eliminated.

2. EXPLORATORY DATA ANALYSIS

Each modelling problem represents a niche aspect of binary image classification, with labels provided in a separate CSV file. The datasets across the three test domains include:

"Is Epic Intro"	"Needs Respray"	"Is GenAI"
Determine from a spectrogram whether a song is "Epic"	Identify weeds present between pavers and label them for respray.	Differentiate between captured photos and images generated by AI.

Table 2.1: Description of the Three Test Domains

2.1 Image Analysis

To analyse the images, we plotted RGB channel histograms of some images, to check whether the images required contrast adjustment[10]. Upon analysis of the histograms, some images had abnormal peaks in the RGB channels. The figure below illustrates peaks(each peak for each channel) for two images under 'Needs respray' dataset:

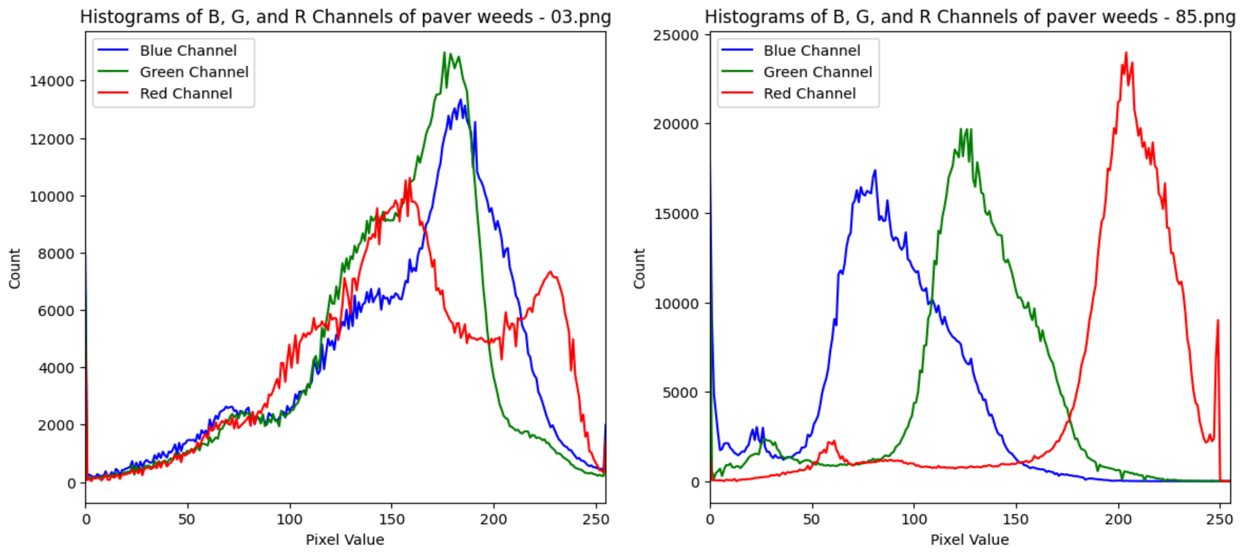


Fig 2.1: Histogram represent the pixel intensity range from 0 to 255

We used the 'Histogram Equalization' method under the **'cv2'** module to spread out the pixel values more uniformly to avoid having abnormal peaks. Further details about this method is given in 'Methodology' under 'table 2.1'.

2.2 Class Distribution

Across all the three domains, we did not apply any class balancing techniques such as oversampling or undersampling as the dataset for the domains was already balanced. The dataset for Needs respray has 6 images for each class, Is epic intro has 5 images each but the Gen AI dataset has 10 images per class. The figure below illustrates the class distribution of the three datasets:

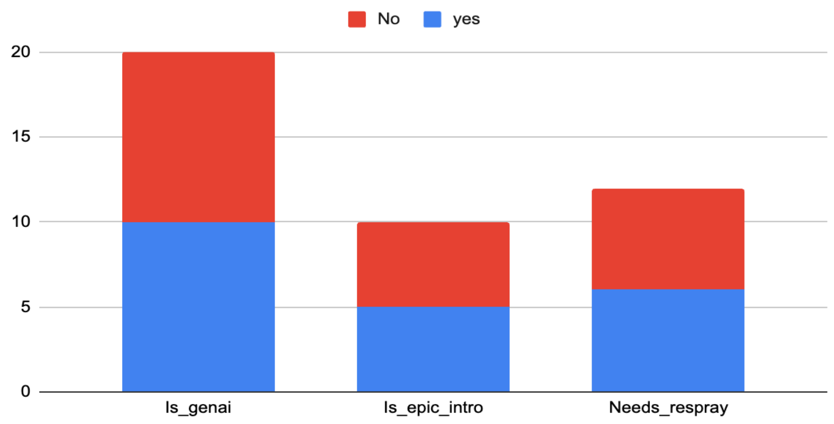


Fig 2.2: Class Distribution of the Three Datasets In a Bar Graph

3. <u>METHODOLOGY</u>

3.1 Pre-processing

Data augmentation is crucial in performing well for any few-shot learning problem, as it is the only method of obtaining adequately sized training sets. In order to best preserve the generalisability of the model we used 17 different augmentation techniques to comprehensively train on every aspect and create as much robustness to human error as possible, and significantly increasing the diversity of the training samples[12][8]. For ease of prediction comparison a label encoder was used to convert string labels into a **1HOT encoded format**. This component is crucial for maintaining consistency in label handling during both training and deployment phases.

Augmentation technique details based on the specific parameters in our code[8]:

Technique - Description	Relevance to dataset
Rotational transform - performs a fixed quarter rotation clockwise using OpenCV `rotate` function.	Make sure the model can make correct classifications no matter the orientation of the image.
Translation - The code applies translation by randomly selecting a shift in the X and Y directions within a range of -50 to 50 pixels.	Teaches the model to recognize objects even when they are partially shifted out of frame.

Zoom (Scaling) - resizes the image by a random scale factor between 0.8 and 1.2.	Effectively simulates zooming in and out, helping the model learn to identify features at various scales.
Flipping - Randomly chooses to flip the image horizontally, vertically, or both.	Enhances the dataset by introducing mirror variations of images, important for symmetry recognition in objects.
Brightness Adjustment - Adjusts the image brightness by a random factor between 0.5 and 1.5.	Ensures the model learns to recognise features regardless of lighting conditions.
Contrast Adjustment - Modifies the contrast by adjusting the image's pixel values using a scaling factor and adding a bias.	simulating variable lighting conditions
Cropping - Randomly crops a portion from the centre of the image, reducing the size by half both vertically and horizontally.	Ensures robustness of the model to incomplete data.
Shearing - Applies a shearing transformation with a factor randomly chosen between -0.2 and 0.2, slanting it horizontally or vertically.	Simulates images taken from different angles, ensuring robustness to perspective changes.
Normalisation - Scales the pixel values to a range of to 1	Stabilises the training process and helps the neural network converge faster.
Gaussian Noise - Adds random noise generated from normal distribution.	Helps the model become robust against various types of noise in real-world scenarios.
Colour Channel Shifting - Swaps the Red and Blue channels of the image.	Teaches the model to not rely heavily on specific colour channel configurations, enhancing robustness to colour variations.
Histogram Equalization - Applied to the grayscale version of the image.	This method enhances the contrast, making the details more prominent which is useful in conditions of poor contrast.

Saturation (HSV space) - Reduces the saturation by 50%, converting the image to HSV colour space and adjusting the saturation channel.	Helps in training models that are robust to variations in colour intensity.
Hue (HSV space) - Shifts the hue by 10 degrees.	Simulate different lighting conditions or camera characteristics, helping the model to generalise across different colour tones.
Blur and Sharpen - Blur: Applies a Gaussian blur with a kernel size of (5, 5). Sharpen: Uses a kernel to enhance edges.	Blur: Simulates the effect of lower resolution or focus, which is common in real-world images. Sharpen: Helps the model to detect edges and features more effectively.
Scale (Resize)	Resizes the image to a fixed size of 500x500 pixels. This standardisation is crucial when training models to maintain consistent input sizes.

Table 3.1: Data Augmentation Techniques

3.2 Feature Extraction

With such a small training set, if trained from scratch, the model would overfit, or memorise, the training data and perform poorly on any unseen data. To circumvent this, features were extracted using the pre-trained VGG16 model, with frozen weights learned from over 14 million images (the ImageNet dataset). By using the optional parameter "**Include_top=False**", the output from the last convolutional layer of VGG16 can be obtained to use as the feature map for our specialised classifiers to learn.

3.3 Model descriptions

- **VGG16-based Classifier**: A custom NN classifier using VGG16 for feature extraction. It consists of 3 fully connected dense layers with 2048, 512 and 2 neurons respectively. These are Interspersed with dropout layers set to 0.5 (50%) to minimise overfitting, thus maximising generalisability. The first 2 layers use the ReLu activation function to speed up convergence, introduce non-linearity appropriate for an image feature map, and avoid issues with vanishing gradients. Using Softmax in the final layer ensures outputs are in the range $y \in [0,\ 1]$.RMSProp was the chosen optimization algorithm, as both the adaptive learning rate and the normalisation of gradients is well-suited to the highly complex feature set and potentially sparse gradient values seen in image classification problems. It was set with a relatively low learning rate of $r = 10^{-4}$ to minimise the risk of overshooting minima.

- **Random Forest**: A Random Forest classifier with 5000 trees is trained on the VGG16 features. The high value chosen for the amount of estimators ensures the model can achieve the complexity needed to learn the highly detailed feature map output of VGG16[16]. This ensemble method is known for its robustness and ability to handle overfitting, which makes it highly suitable for learning from a smaller training set with high dimensionality. By using a fixed random state of 42, reproducibility is ensured.
- **Support Vector Machine (SVM)**: An SVM with a **linear kernel** is used, suitable for high-dimensional data like image features. Although the feature space may not necessarily be linearly separable, the model performed worse when using the RBF kernel, so we opted for the better performing option. Furthermore, by using a linear kernel, the high risk of overfitting due to few training samples is less than that of using more complex kernels like RBF or polynomial.
- **K-Nearest Neighbors (KNN)**: A KNN classifier with cosine distance metric, specifically advantageous for its simplicity and effectiveness in handling cases where the decision boundary is very irregular. By using K = 3, emphasis is placed on the local data structure, suitable for the highly dimensional feature space of image data as they will be well-separated. Instead of the Euclidean distance metric, which is heavily affected by the curse of dimensionality, the Cosine distance metric provides a more meaningful metric when dealing with high-dimensional spaces[3].
- **Decision Tree**: A single Decision Tree classifier provides a baseline comparison. It offers clear visualisation of decision-making and high interpretability, but is likely to overfit on complex image data. As with the RF model, setting "random_state=42" ensures reproducibility and consistency between instances.
- **DenseNet-Based Classifier**: instead of using VGG, [11] the Densenet Convolutional layers were frozen, preserving the pre-trained feature extraction capability of the model. The model then uses global average pooling followed by a dropout layer and a dense layer for binary classification, optimised using the Adam optimizer and early stopping based on validation loss to prevent

overfitting. The Adam optimiser is well suited for image classification, as the adaptive learning rate eliminates the risk of encountering vanishing gradients. The choice of binary cross-entropy as the loss function for a binary image classification is a given.

3.4 Model Aggregation

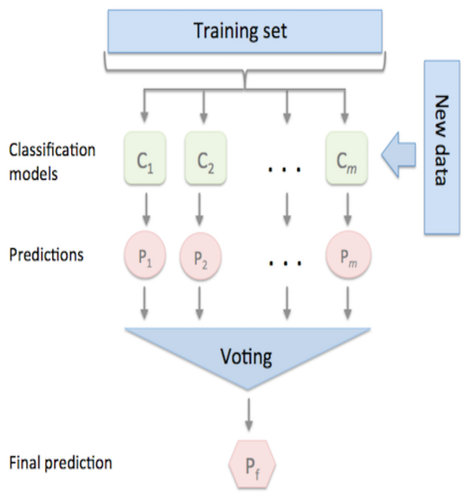


Fig 3.4: Majority Voting

Individual models may have unique weaknesses or may make different errors, however when their predictions are combined through majority voting, the unique approach that each model brings to the problem can be leveraged for increased generalisability to a diverse problem set. Different models intrinsically approach the provided feature map in different ways and have different biases. These biases will be very pronounced due to the low availability of training data, and thus by using a voting system from multiple models, ideally the bias will be minimised.

4. RESULTS

Due to the absence of a conventional test set, we generated a dummy dataset [17] (Skin Cancer Binary Classification Dataset) that simulates the variability of unseen data, testing the performance and robustness of the model under various conditions. In this section, we analyse each of the performance metrics; accuracy, precision, recall, and F1-score across each subset of data. For the project, we divide the training dataset into two distinct subsets: the training set, encompassing 80% of the total samples, and the validation set, constituting the remaining 20%.

4.1 Feature Extraction

While VGG-16 provides features for Random Forest, Decision Tree, KNN and SVM, DenseNet directly processes the input and augmented images for feature extraction. DenseNet's direct image processing contrasts with VGG-16's pre-trained feature utilisation, offering a unique approach. By leveraging both direct input and feature extraction, a more generalisable model is hopefully obtained.

4.2 Evaluation Metrics

Although values for 5 different metrics were obtained, we focused mainly on F1 score as it encapsulates the model's ability to achieve a better combination of precision and recall. For evaluating loss at each epoch, Binary CrossEntropy Loss was used as it penalises the model based on the distance between predicted probabilities and actual labels. The definition of Binary Cross-Entropy Loss is:

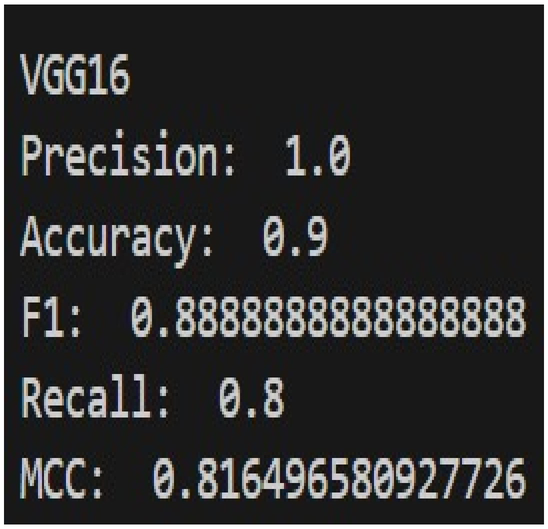
$$Binary\ CrossEntropy\ Loss = \ - 1/n \sum_{i=1}^{n} (y_i(log(p_i) + (1 - y_i)log(1 - p_i))$$

Equation 1: Binary CrossEntropy Loss

Where, "n" depicts the total number of samples, "y_i" denotes the true label of the ith sample (0 or 1), and "p_i" represents the predicted probability that the ith sample belongs to the positive class (class 1).

4.3 Model Selection [Majority Voting]

As discussed under **"3.4 - Model Aggregation"**, We combined the predictions of VGG16 with custom dense layers through the majority voting method, VGG16 scored better on the F1 metric compared to using other models for feature extraction. The below graphs (from 4.3.1 onwards) represent VGG16 performance on three different datasets, please see appendix for a complete list of confusion matrices. Testing VGG16 with custom dense layers used on a dummy dataset gave the following results and the graph:



```
VGG16
Precision:  1.0
Accuracy:  0.9
F1:  0.888888888888888
Recall:  0.8
MCC:  0.816496580927726
```

Fig 4.3.1 Confusion Matrix

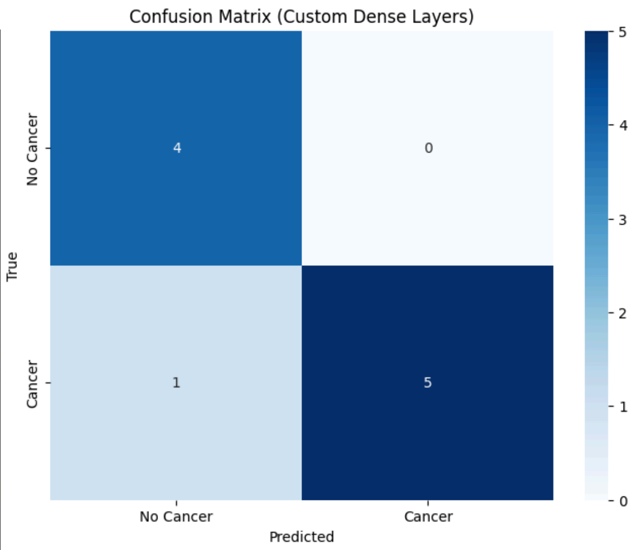


Fig 4.3.2 Model Performance

4.3.1 GenAI -

The right graph shows diminishing training and validation loss, with the latter declining at a slower pace, which indicates model convergence. On the left, both training and validation accuracies exhibit upward trends, indicating steady improvement without overfitting. These patterns illustrate the model's capability to learn from minimal training data while retaining its effectiveness in generalising to new unseen examples.

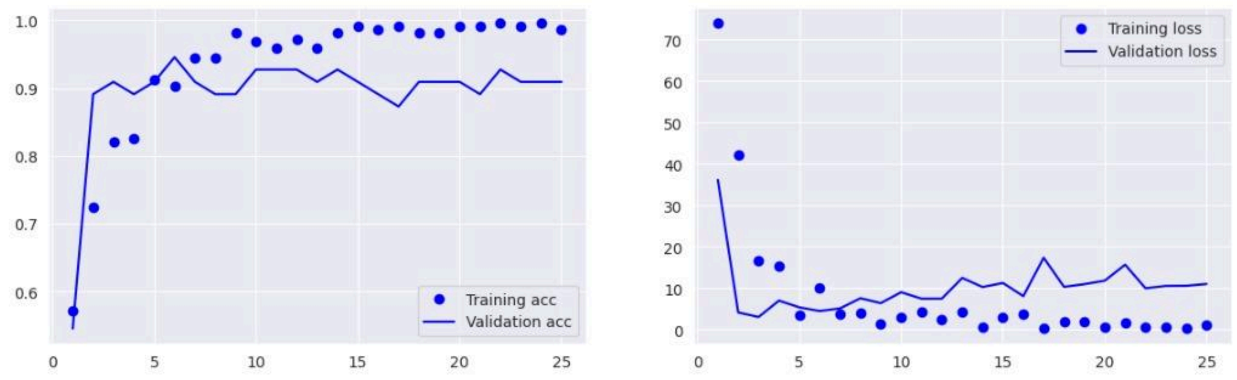


Fig 4.3.1.1 GenAI

4.3.2. Is Epic -

In the left graph, the training accuracy increases with model training over the epochs, but a slower increase in validation accuracy. On the right, the training loss decreases as the model learns from the training data, while a less significant decrease in validation loss, where the model fails to generalise effectively to new data. These highlight the balance between model learning and avoiding overfitting, crucial for optimal performance in machine learning tasks.

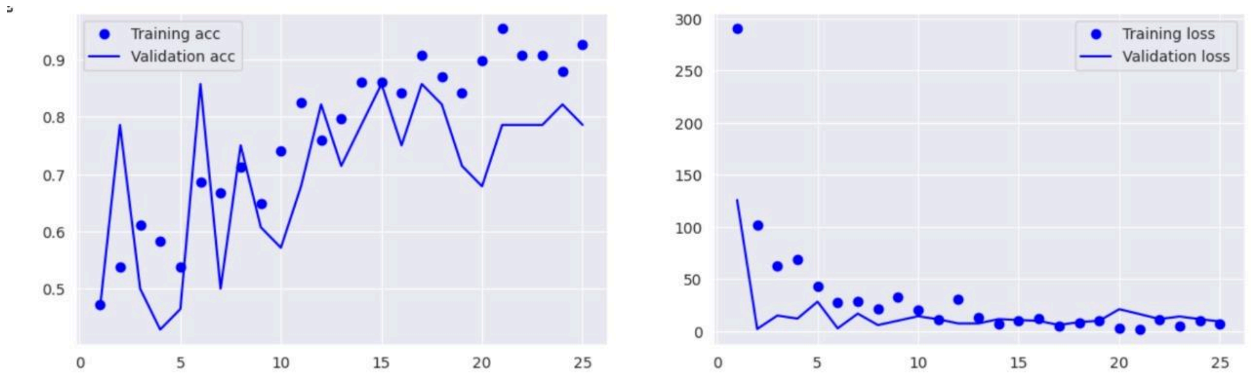


Fig 4.3.2.1 Is Epic

4.3.1.3 Need Respray -

In the left graph, the increasing training accuracy as the model significantly improves its fit to the training data over epochs. Whereas, the right graph depicts the training loss gradually decreasing as the model learns from the training data over the time However, a slower increase in validation accuracy may indicate overfitting. These observations underline the importance of monitoring both loss and accuracy metrics to ensure model performance and generalisation capability.

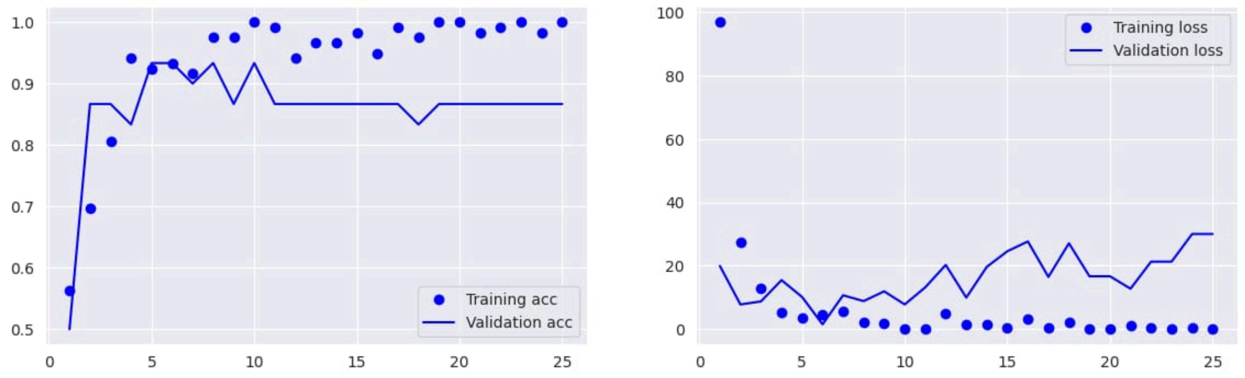


Fig 4.3.3.1 Need Respray

4.4 Model Comparison

The evaluation results presented in the table showcase the performance metrics of various models tested on a dummy dataset. VGG16 with custom dense layers depicts satisfactory accuracy and F1 score, which indicates robustness among classification. Decision Tree and Random Forest models show better accuracy, while Decision Tree highlights higher precision. Particularly, Random Forest demonstrates perfect precision and worthy recall, which signifies reliable predictions across classes. SVM and KNN models maintain consistent precision, recall, and MCC values. Even with little less precision, DenseNet maintains competitive accuracy and F1 score.

Model	Accuracy	F1 Score	Precision	Recall	MCC
GG16 + custom denselaye	0.90	0.88	1.0	0.80	0.82
Decision Tree	0.87	0.81	0.88	0.8	0.81
Random Forest	0.9	0.75	1.0	0.8	0.82
SVM	0.8	0.80	1.0	0.6	0.65
KNN	0.8	0.75	1.0	0.6	0.65
DenseNet	0.80	0.83	0.71	1.0	0.65

Table : Overall Comparison of the Model's Performance'

5. DISCUSSION

5.1 Comparison of different methods, their features, and performances

In our comparative analysis of features, VGG16 was selected as the primary feature extractor due to its superior well-documented benchmark performance, and our experimentation using other feature extractors like VGG19. From the six models analysed, Decision Tree had the lowest performance, attributed most likely to the prohibitive simplicity of the model. Random Forest improved on this with better accuracy obtained through an ensemble approach more able to capture complexity. SVM and KNN showed high precision but struggled with recall, indicating challenges in identifying all positives. DenseNet stood out with excellent recall, proving efficient in feature utilisation but had some false positives. Performance wise, Custom Dense Layers and Random Forest were best, while Decision Tree showed moderate results. SVM and KNN aligned with their predicted limitations. DenseNet, despite having slightly lower precision, was strong in recall ability, highlighting its potential to capture essential details in images.

5.2 Discussion of the appropriate metrics

While accuracy provides an overview of correct predictions, F1 score is a more detailed metric, using more of the confusion matrix. [15] MCC is also considered an important metric, given its comprehensive nature in evaluating binary classification, ensuring that all aspects of the model's predictive capabilities are taken into account.

$$MCC = \frac{TP\ X\ TN - FP\ X\ FN}{\sqrt{(TP + FP)\ (TP + FN)\ (TN + FP)\ (TN + FN)}}$$

Equation 2: Matthews Correlation Coefficient (MCC)

5.3 Discussion of Future Improvements

While the models currently used are performing well, there's always room for growth. Looking ahead, here are some ways we could make them even better:

- **Few-Shot Learning**: [4] Few-shot learning strategies can be explored, adapting pre-trained models to our specific tasks with only a few labelled examples. This could help overcome the overfitting often associated with small datasets.
- **Data Augmentation and Generation**: Expanding our dataset through advanced data augmentation and generative models, such as GANs, could provide our models with more varied examples to learn from, which leads to improving their generalisation capabilities.
- **Model Tuning:** The models' hyperparameters could be fine-tuned further to improve performance.

6. CONCLUSION

In conclusion, we tackled the challenge of few-shot image classification by developing a highly generalizable AI model capable of differentiating between classes with minimal training data. By combining exploratory data analysis with advanced data augmentation and an ensemble of models including VGG16, DenseNet, SVM, KNN, and Random Forest, we achieved significant accuracy despite the data constraints. Our results, validated by metrics such as accuracy, F1 score, and MCC, demonstrated the effectiveness of our methodologies. Future improvements could involve integrating few-shot learning strategies such as meta-learning, advanced data augmentation techniques, and further tuning of model parameters to enhance performance to ultimately having a generalizable classifier in scenarios with limited data availability.

7. REFERENCES

[1] Vipul, K. (2021). Build our First Image Classifier With Convolutional Neural Network (CNN). *Towardsdatascience*. Retrieved from: https://towardsdatascience.com/build-our-first-image-classifier-with-convolution-neural-network-cnn-b4e9034ec5c

[2] Ankit, P. (2018). Understanding our Convolution network with Visualizations .*Towardsdatascience.* Retrieved from: https://towardsdatascience.com/understanding-our-convolution-network-with-visualizations-a4883441533b

[3] Tarek, H. (2022). Similar image: CNN + Cosine Similarity. *Kaggle*. Retrieved from: https://www.kaggle.com/code/hamditarek/similar-image-cnn-cosine-similarity/notebook

[4] Clement, W. (2022). Image Classification with No Data? *Towardsdatascience.* Retrieved from: https://towardsdatascience.com/image-classification-with-no-data-b44089f8dc28

[5] Bekhzod, O. (2024). Spectrogram Images Visualization and Classification. *Kaggle*. Retrieved from: https://www.kaggle.com/code/killa92/spectogram-images-visualization-and-classification

[6] Ashit, A. (2024). Spectrogram Image Classification. *Kaggle*. Retrieved from: https://www.kaggle.com/code/ashitaggarwal1986/spectogram-image-classification

[7] Olga, C. (2021). Complete Guide to Data Augmentation for Computer Vision. *Towardsdatascience.* Retrieved from: https://towardsdatascience.com/complete-guide-to-data-augmentation-for-computer-vision-1abe4063ad07

[8] Wei, S. (2023). Exploring Different Image Augmentation Methods in TensorFlow/Keras. *Medium*. Retrieved from: https://medium.com/@shouke.wei/exploring-different-image-augmentation-methods-in-tensorflow-keras-e2e9b3e94b#34e1

[9] Vasani, D. (2019). Sound classification using Images, fastai. *Towardsdatascience.* Retrieved from: https://towardsdatascience.com/sound-classification-using-images-68d4770df426

[10] Rausch, D. (2021). EDA for Image Classification. *Medium*. Retrieved from:
https://medium.com/geekculture/eda-for-image-classification-dcada9f2567a

[11] Martins, C. (2023). DenseNet Architecture Explained with Code Examples.
Medium. Retrieved from:
https://cdanielaam.medium.com/densenet-architecture-explained-with-code-examples-0d86d7936f83

[12] Shorten, C., Khoshgoftaar, T.M. (2019). A survey on Image Data Augmentation for Deep Learning. J Big Data 6, 60. Retrieved from: https://doi.org/10.1186/s40537-019-0197-0

[13] Sarojag. (2024). Know About Zero-Shot, One-Shot, and Few-Shot Learning. *Analytics Vidhya*. Retrieved from:
https://www.analyticsvidhya.com/blog/2022/12/know-about-zero-shot-one-shot-and-few-shot-learning/

[14] L. Brejon. (2021). Extract Features Models [Notebook]. *GitHub.* Retrieved from:
https://github.com/lbrejon/Compute-similarity-between-images-using-CNN/blob/main/notebooks/extract_features_models.ipynb

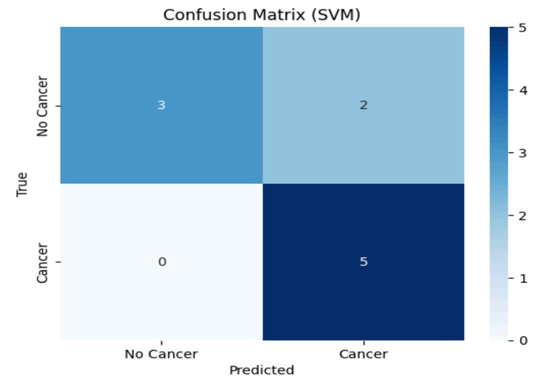
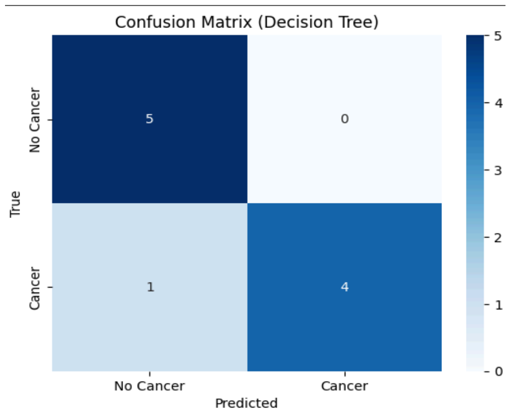
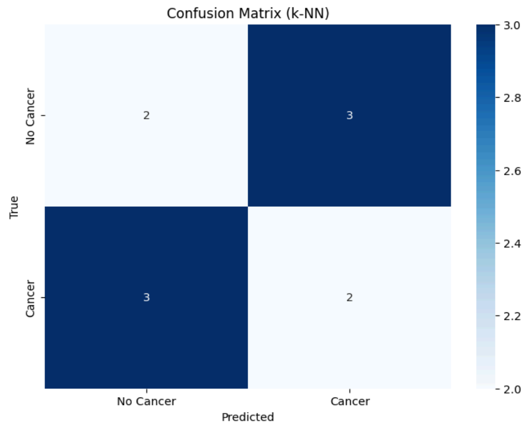
[15] Li Sisters. (2020). Matthews Correlation Coefficient: when to use it and when to avoid it. *Towardsdatascience.* Retrieved from:
https://towardsdatascience.com/matthews-correlation-coefficient-when-to-use-it-and-when-to-avoid-it-310b3c923f7e#:~:text=class%20is%20positive.-,Conclusion,metric%20for%20binary%20classification%20problems

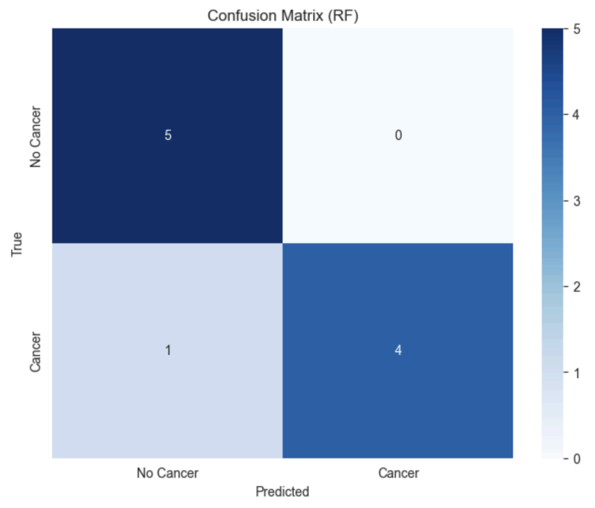
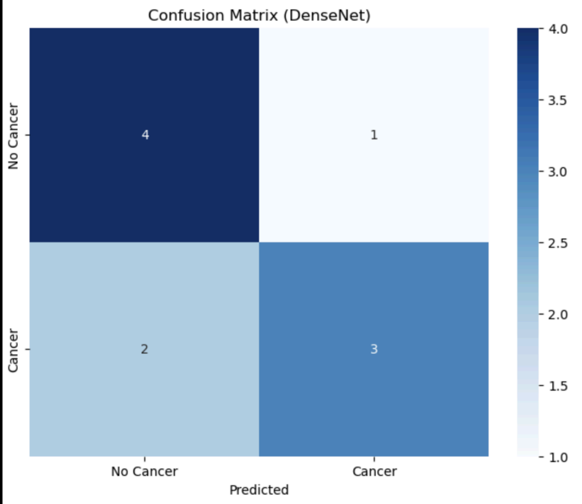
[16] scikit-learn. (n.d.). RandomForestClassifier. *Scikit-learn.* Retrieved from:
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#:~:text=A%20random%20forest%20is%20a,accuracy%20and%20control%20over-fitting.

[17] Graupe, K. (2023). Skin Cancer Binary Classification Dataset. *Kaggle*. Retrieved from:
https://www.kaggle.com/datasets/kylegraupe/skin-cancer-binary-classification-dataset

8. APPENDIX

8.1 Confusion Matrix

SVM	
DECISION TREE	
k-NN	

RANDOM FOREST	
DENSENET	 Confusion Matrix (DenseNet) No Cancer Cancer 4 1 2 3 No Cancer Cancer Predicted 4.0 3.5 3.0 2.5 2.0 1.5 1.0