CSC8635 Main Report

Student Number: 180322262

Abstract

This projects, focuses on models which can identify, types of skin cancer. Making a correct identification, is extremely important, since skin cancer is one of the most common types of cancer and is highly treatable if detected early. So, the aim of the project is to find models which can perform image classification, for skin cancer images. Finding accurate models is key as they can lead to faster diagnosis.

The method followed, was to train three models with grayscale and colored images and assess their performance. The models chosen, are the CNN, SVM, SVM + LDA, and RF + LDA. All were trained using 28x28 images. For evaluation, the factors considered were the testing accuracy, the computational time, and the advantages/disadvantages of each model.

Results showed that the model with the lowest computational time was the RF + LDA model on colored images, which took approximately 112s to train and gave a testing accuracy of around 92%. The SVM model fitted on grayscale images had the best accuracy of 98%, but the highest computational time (673s), while the SVM + LDA on colored images had an accuracy of 97% and a computational time of 578s. Finally, the CNN for grayscale and colored images had 75% and 95% accuracy and 178s and 383s training time respectively.

The choice of the models, and the combination of techniques have been used before for image classification. However, for this dataset, the most common model used for classification is CNN. This project investigates if there are other models that can be more accurate and efficient.

Key images

Model	Testing Accuracy	Time (seconds)
CNN (grayscale)	75%	178.7 s
CNN (RGB)	95%	383.88 s
SVM (grayscale/sample)	98%	673.24 s
SVM + LDA(RGB)	97%	578 s
RF + LDA(RGB)	92%	112 s

Project Background

The data for this project consists of approximatelly 10000 images of seven different skin cancer types. The different types are Melanocytic nevi (nv), Melanoma (mel), Benign keratosis-like lesions (bkl), Basal cell carcinoma (bcc), Actinic keratoses (akiec), Vascular lesions (vasc), and Dermatofibroma (df). These images have been encoded to pixels and have been resized to be 8x8

and 28x28. Moreover the data set consists of both grayscale and colored images which are used for the project in order to see whether grayscale or colored images produce more accurate results. An example of the original images used for the project is found below:

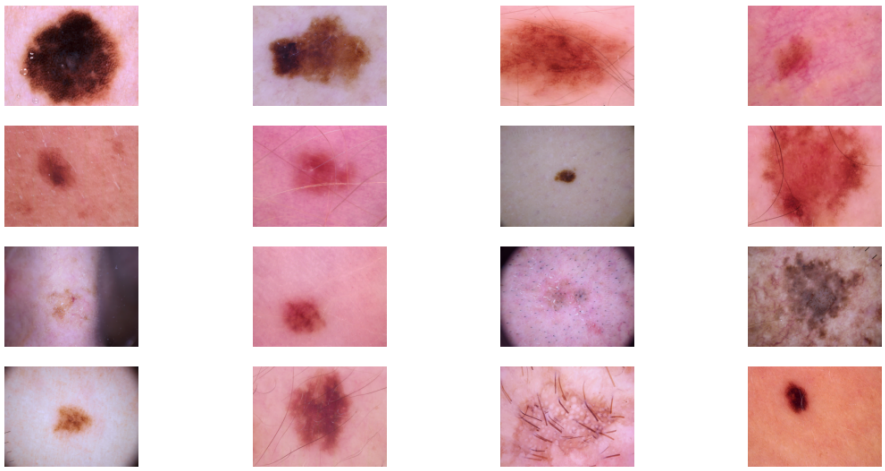

Figure 1: Figure showing some examples of the images used to train and test the models

The images of the datasets are used as a training and test set for building an image classification model and the aim of the project is to find the model which will be able to classify images as accuratelly as possible. However, there are more to this aim. For example, in order to choose the final model, the computational time will be taken into account. So when compariung to models with similar accuracy, the time needed to train the model will be taken into account.

Before moving on with the work undertaken for this project it is very important to get a better idea of the dataset which is used for the project. For this reason some main results are visualised in the plots below.

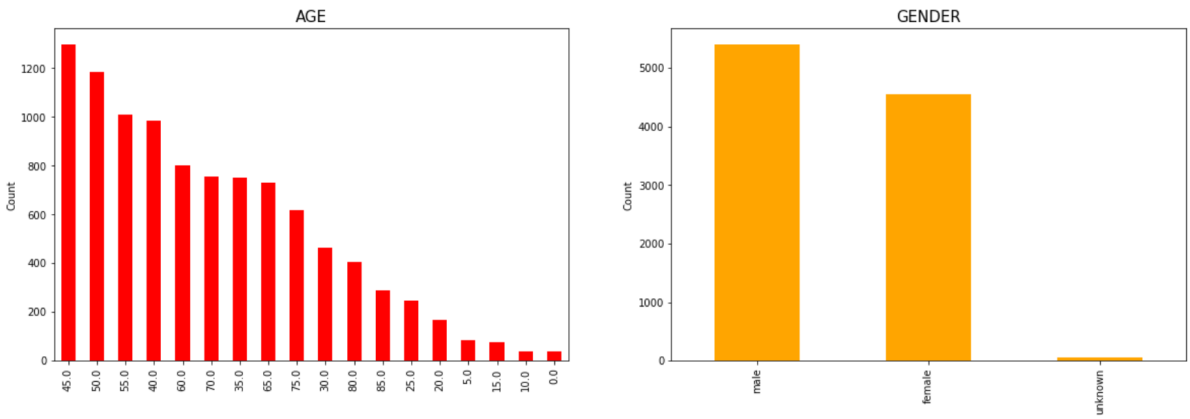

Figure 2: Bar plots showing the age count (left) and gender count (right) in the dataset

From the right plot above, it is clear that there was a similar number of male and female patients,

and a very low number of patients with unknown gender. So there is no reason of concern, meaning that there are used skin cancer examples from both genders when training the models. When it comes to age, it seems like in this dataset the majority of the image samples come from people that are between 40 and 55 years old, while it seems like there are not many samples from younger people as expected . Another area to look at is which, if any, skin cancer types appear to be more frequent. Moreoever it is interesting to investigate if there are locations in the body where cancer appears more often.

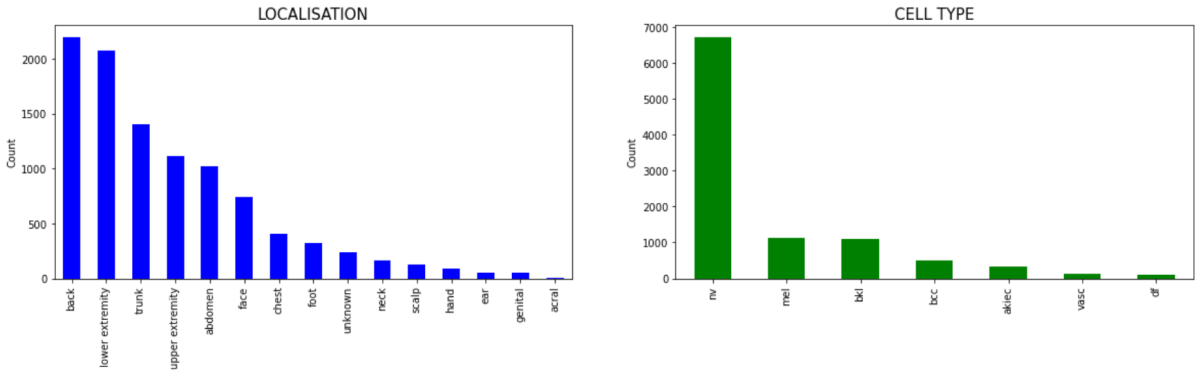

Figure 3: Bar plots showing the localisation count (left) and cell type count (right) in the dataset

From the left plot above it is clear that most of the skin cancers in this dataset are located in the back or the lower extremity, while skin cancer located in the ears, genitals, or acrals is much less common. Moreover it should be noted that there are also some skin cancer images where the location of the cancer was not stated. Looking at the right plot it is very clear that melanocytic nevi is the cancer type which appears more often in this dataset, while dermatifibroma is the type which appears the least. So it is clear that the data are imbalanced and oversampling may need to be performed before training any model. Having looked at all the above plots, another detail to look at is whether age plays a role in the type of cancer. The plot below visualises the results.

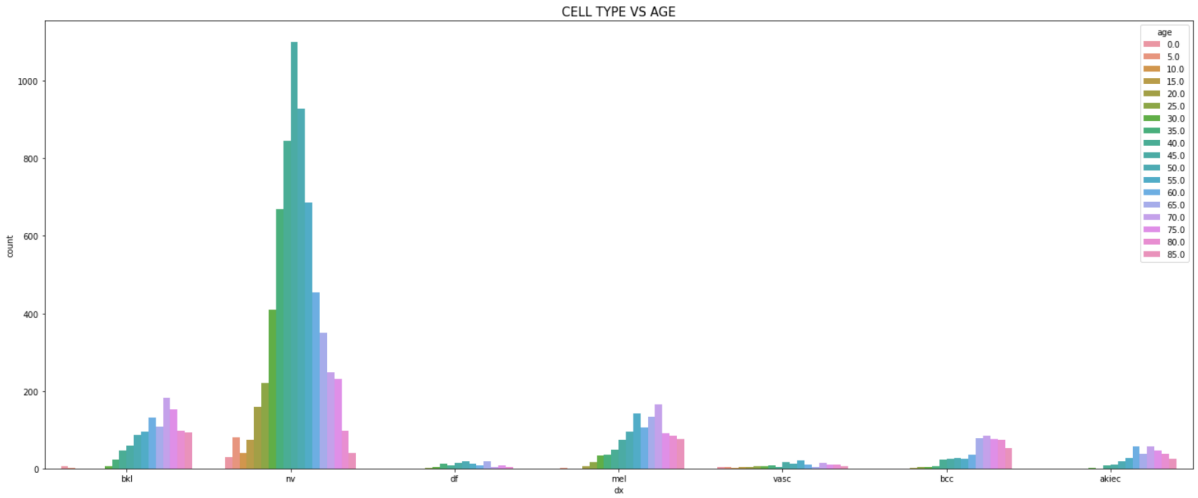

Figure 4: Plot of cell type count by age

From the plot it is clear that melanocytic nevi which is the type which appears more ofter in the

data set is more common in younger people aged from 35 - 55, while all other types seem to be more common in people aged 65 and older. Finally for recognising the type of cancer, around 50% of lesions were identified through histopathology. In order to recognise the rest of the cases follow-up examination (followup), expert consensus (consensus), or confirmation by in-vivo confocal microscopy was used. The results are summurized in the pie below.

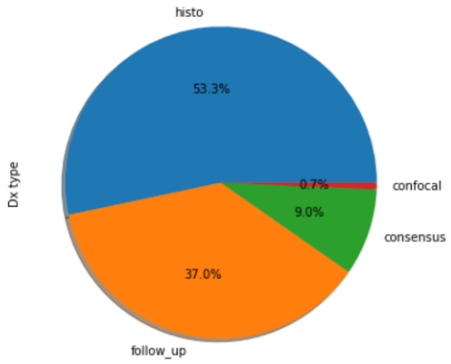

Figure 5: Pie chart showing the identification methods for the samples in the dataset

Aim and plan

As mentioned above the aim of the project is to build an efficient and accuarate model for image classification for these specific cancer types. There are many machine learning models some of which are supervised and some of which are unsupervised. For this specific project, the data are already labeled so only supervised models were fitted. The models chosen for this project are CNN, SVM, and Random Forest classifier with LDA. These were chosen as they seemed to be appropriate for image classification, since there was a lot of bibliography on this.

The plan followed for this project is quite structured. The first step taken was understanding the data and how they can be used for the project. Two main differences in the csv files were identified. The first difference was that there were two files with 8x8 images and two with 28x28 images. For this project only the 28x28 files were used because there is evidence that higher resolution images lead to more accurate models [(Thambawita et al., 2021]. For the CNN and SVM models both the grayscale and RGB images were used in order to compare the accuracy of the models as well as the computational time needed to train the model.

The reason for choosing to fit the CNN model is that it is one of the most widely used algorithms for machine learning image classification [Tripathi, 2021]. So it was interesting to see if the model was appropriate for this dataset. Then for the SVM again both grayscale and colored images were used. However the computational time for this model was extremelly high, when using the whole set of training images, so a subset was selected. To deal with this problem LDA was combined with SVM, and the whole dataset was used. The results are analysed in the next section. There is bibliography on this topic, where SVC was combined with PCA [Hasan et al., 2019], as well as LDA and the results were really good [Cheng and Chia Li, 2010]

The final model chosen was the Random Forest classifier with LDA. So, firstly the dimension of the data was reduced using LDA, and then the model was fitted. This combination was chosen in order to reduce the time needed to train the model. It has been applied before for face recognision, where LDA was used for feature extraction and then Random Forest as a classifier. LDA has also been used before, combined with other machine learning algorithms. Moreover, there are evidence that the Random Forest Classifier with the whole dataset provides similar accuracy to that of when using the dataset after performoning dimension reduction with LDA [Ching Chen et al., 2020]. So that was the main reason why this last model was chosen. All of the models chosen have advantages and disadvantages which are discussed below.

Model Fitting

Before fitting any model some specific steps were followed. As mentioned above the data for this project are quite imbalanced. This is clear from the plot below.

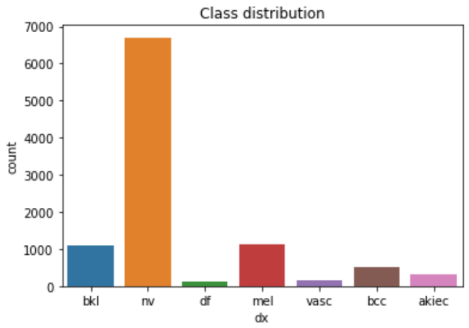

Figure 6: Bar plot showing the representation of each class in the dataset

It is evident that the nv class is over represented in the data set.So, the first thing to do was to over sample the data. This was done by using the SMOTE algorithm. There is evidence which suggests that for specific models like the CNN and SVM the performance is worse when the dataset used is imbalanced [Hensman and Masko, 2015]. Moreover it was found that Random Forest classifier performs well, even with imbalanced data. However in the specific project, dimension reduction using LDA is performed before fitting the RF model. Evidence shows that LDA performs better with balanced data, so again, when reducing the dimensions, the data is first balanced [Tischio and Weiss, 2019]. However, it should be mentioned, that over sampling can lead to overfitting in the models, so the accuracy of some models may be better due to overfitting. The second step to take was to split the data into training and test set, in order to use a portion of the data for training the model and then the rest of the data to test it. This was done using the train test split algorithm. For this project 80% of the data were used for training and 20% for testing. The third step taken, was to normalise the data, by dividing each pixel by 255, in order to bring all data in the [0,1] range. This is done in order to make the computation in the models easier and faster. There are other ways for handling the data such as standardisation, and testing if there is a way that works better for a specific dataset, could be an extra step to take. This is not done for this project, but it would definitely be something to consider for future projects, or for this specific one.

The steps above were taken when fitting all of the models. However, for specific models, some extra steps were taken in order to reshape the data in the appropriate input dimensions. All the extra steps are discussed for every model below.

CNN model

CNN for grayscale images

As mentioned above the first model fitted was the CNN model. In order to fit the model the steps described obove were implemented. In addition to these steps, the data were reshaped to 28x28x1, in order to have the right input dimensions. The CNN was fitted using both the grayscale images as well as the the RGB images. For this model the 28x28 images were used. The initial model fitted included two dense and two dropout layers. The kernel size was 3x3 and the activation functions were relu and softmax. Moreover the optimizer used was adam. Finally the batch size used was 128 and the number of epochs was 20.

The training accuracy for this model was around 69%, the validation accuracy was around 67% while the testing accuracy was approximatelly 75%. The time needed to fit this model was 178.7 seconds. From the testing accuracy it seems like 75 out of 100 images were guessed correctly. The test accuracy seems to be a bit higher than the training accuracy. This is a bit concerning, as it could mean that the model is overfitting. However the difference is not that large. The accuracy and loss plots are found below.

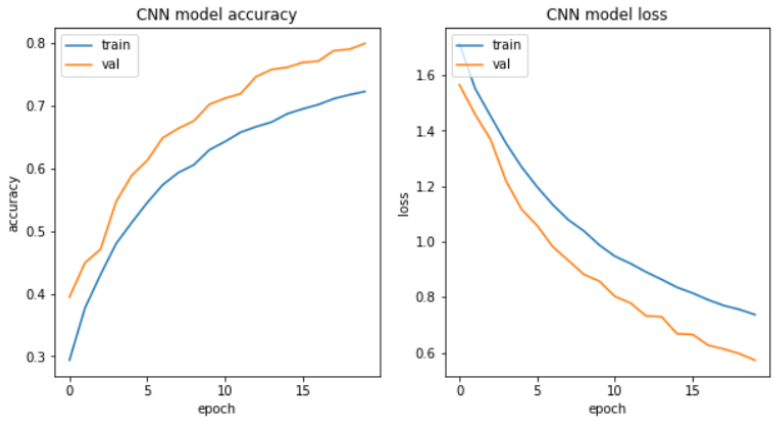

Figure 7: Graphs of training and validation accuracy for CNN model using grayscale images and balanced dataset

However, when fitting the CNN model without oversampling, the training accuracy is 70% and the testing accuracy is 69%. So now the training accuracy is very close to the testing one, which is a better indication. Compared to the CNN model, with the balanced data, it is observed that the testing accuracy is lower here, but loss is higher. Another difference is that the elapsed time for the imbalanced data model, is just 39.5 seconds, which is considerably lower. Nevertheless, it should not be forgotten that using the imbalaced dataset, may lead to problem because the model may spend most of the time learning about the class which has the more samples. In this case, however, it seems that the model performs okay even with imbalaced data. The accuracy and loss plots are below:

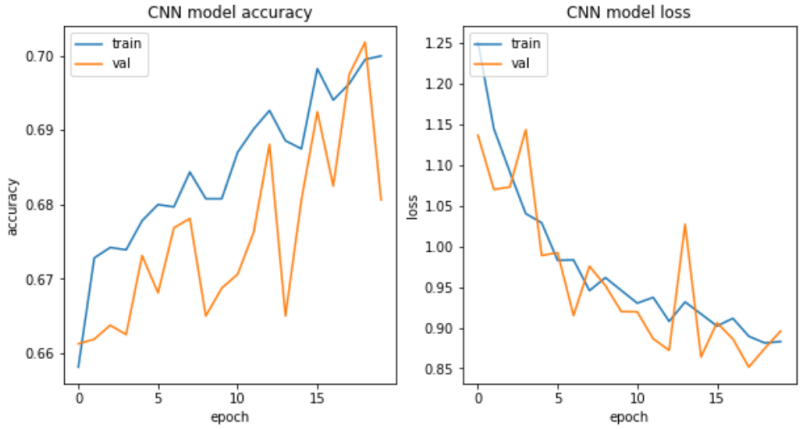

Figure 8: Graphs of training and validation accuracy for CNN model using grayscale images and imbalanced dataset

CNN for RGB images

After having fitted the CNN model to the grayscale images, it is time to fit it to the RGB images. Again the steps described above are followed, but this time the data are shaped into 28x28x3, because of the three different colors. The kernel size is again 3x3 and the activation functions are the same.The bach size and the number of ephocs are the same as before. However, here three dense layers are added since the input dimensions are higher. For this model oversampling is performed. After fitting the model, the training accuracy, is approximatelly 97% and the test accuracy is approximatelly 95%. Simalrly the training loss is around 0.07, while the test loss is 0.21. The plots below show the accuracy and loss for train and validation sets.

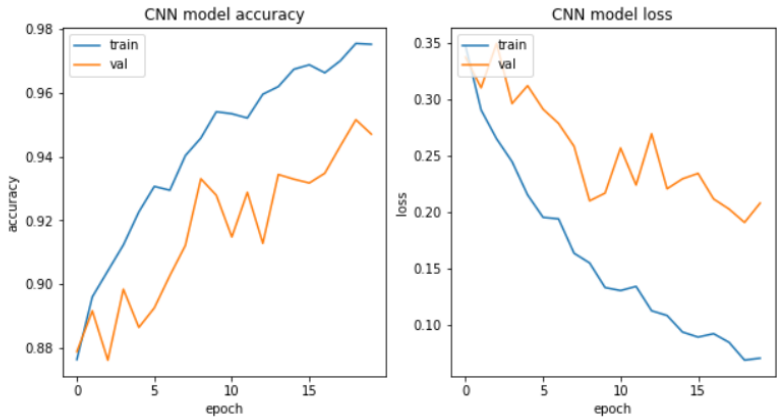

Figure 9: Graphs of training and validation accuracy for CNN model using colored images and balanced dataset

The time needed to train this model is approximatelly 383.88 seconds, which is considerably higher than the model when the grayscale images are used. It can be noted that if only 10 epochs are used the training time is approximatelly 190 seconds and the test accuracy is around 85%.

To conlcude, when comparing the cnn model for grayscale and RGB images it is clear that the model for the RGB images perform better. However it does take more time to train. In general it seems that the CNN model is quite good for this specific dataset, especially if the RGB images are used.

SVM

The second model fitted was SVM. This model was chosen because it seems to be a really good model for image classification and also quite fast. The steps mentioned above were again followed here. However, here the images are not reshaped, as the input dimensions used to train the model are of the type (number of samples, number of features). Again here the 28x28 images are used and the model is fitted both to the grayscale and the colored images in order to compare performance. Moreover, for this model GridSearchCV is used in order to loop over possible values of the hyperparameters C and gamma. The kernel chosen is the rbf because there is evidence suggesting that for multiclass image classification the rbf kernel is quite appropriate [Fauvel et al., 2006], and in general is one of the most used kernels for SVM. However, one of the drawbacks is that it can be computationally expensive for a high dimensional input space [Prasanna and Don, 2018].

The SVM model was difficult to fit to the whole data set, as the time required to run the GridSearchCV was very high. For this reason, 2000 data were selected randomly, and then oversampling was performed in order to have equal samples from each class.

SVM for grayscale images

Firstly the grayscale images were used. After GridSearchCv the optimal C was found to be 10 and the optimal gamma was found to be 0.5. This gave a test accuracy of approximatelly 98% which is very good. However a drawback of this method was that the GridSearchCV took approximatelly 632 seconds (11 minutes) and this was on a train set of shape (7487, 784). So if the whole dataset was used after oversampling then the time needed for GridSearchCV would be much higher. The results on GridSearchCV are shown below.

```
# Tuning hyper-parameters
The score is 0.9868697478991597
best parameters from train data:  {'C': 10, 'gamma': 0.5, 'kernel': 'rbf'}
-------------------------------------
0.872 (+/-0.015) for {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}
0.975 (+/-0.007) for {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
0.983 (+/-0.007) for {'C': 10, 'gamma': 0.5, 'kernel': 'rbf'}
0.957 (+/-0.010) for {'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}
0.977 (+/-0.006) for {'C': 100, 'gamma': 0.1, 'kernel': 'rbf'}
0.983 (+/-0.007) for {'C': 100, 'gamma': 0.5, 'kernel': 'rbf'}
0.965 (+/-0.006) for {'C': 1000, 'gamma': 0.01, 'kernel': 'rbf'}
0.977 (+/-0.006) for {'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
0.983 (+/-0.007) for {'C': 1000, 'gamma': 0.5, 'kernel': 'rbf'}
Time elapsed:  673.6411557197571
```

Figure 10: Figure showing the SVM accuracy for different hyperparameters, for a sample of grayscale images

So in general it seems that SVM is very good for this data set. However the fact that it was fitted on a sample of the training set may lead to overfitting, and so the accuracy may be higher.

The main disadvantage for using this model for the specific dataset, is that the time required to run gridsearchcv in order to tune the hyperparameters is very long compared to CNN, even for a sample of the data set, so it would be even longer if the whole dataset would be used. So it might not be very appropriate for this dataset.

SVM for colored images with LDA

An attempt of following the same procedure as before, was made. However, the time needed for GridSearchCV was too high, since the dimensions of the training set were higher. For this reason SVM was fitted to the whole dataset. In order to make the computational time faster, the dimensions of the dataset were firstly reduced with LDA. So the training set reduced down from (37548, 2352) to (37548, 6). Research indicates that SVM combined with LDA gives really good results for classification [Xiong and Cherkassky, 2005] Besides good accuracy, the computational time is much faster, when compared to fitting SVM to the raw data [Chen and Chia Li, 2009]. Again for this model, the rbf kernel was used. The results of this method are shown below.

```
# Tuning hyper-parameters
best parameters from train data:  {'C': 10, 'gamma': 0.5, 'kernel': 'rbf'}
-----------------------------------
0.943 (+/-0.004) for {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}
0.952 (+/-0.007) for {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
0.978 (+/-0.004) for {'C': 10, 'gamma': 0.5, 'kernel': 'rbf'}
0.944 (+/-0.005) for {'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}
0.960 (+/-0.005) for {'C': 100, 'gamma': 0.1, 'kernel': 'rbf'}
0.978 (+/-0.004) for {'C': 100, 'gamma': 0.5, 'kernel': 'rbf'}
0.945 (+/-0.004) for {'C': 1000, 'gamma': 0.01, 'kernel': 'rbf'}
0.966 (+/-0.005) for {'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
0.978 (+/-0.003) for {'C': 1000, 'gamma': 0.5, 'kernel': 'rbf'}
Time elapsed:  598.4057321548462
```

Figure 11: Figure showing the LDA + SVM accuracy for different hyperparameters, for all the colored images

It is clear that the optimal C is 10 and optimal gamma is 0.5. This combination gives accuracy of approximatelly 98%. Moreover the time needed to perform LDA is GridSearchCV is 598 seconds (10 minutes). So when compared to the previous SVM fit, not only is the computational time less and the model is fitted to the colored images, but also the model is now fitted to the whole dataset, making the result more accurate. When comparing the accuracy, there is almost zero difference, making this model even more preferrable.

When compared to CNN for colored images, this model gives a better accuracy, but the computational time is around 3 times higher for the LDA + SVM model.

Random Forest with LDA

The last model chosen was the Random Forest Classifier, combined again with LDA. As mentioned before there is evidence suggesting that such combination gives quite good results. The reason why RF was chosen is because it is shown to be good for image classification, since it results in high accuracy and also overfitting is less of a problem because all predictions are averaged. However, one of the challenges with this model, is that it is difficult to choose the

number of estimators. In this project in order to choose the number of parameters a loop was created, where the Random Tree Classifier, was fitted to each number of estimators, and then the accuracy was calculated. The results are shown in the plot below.

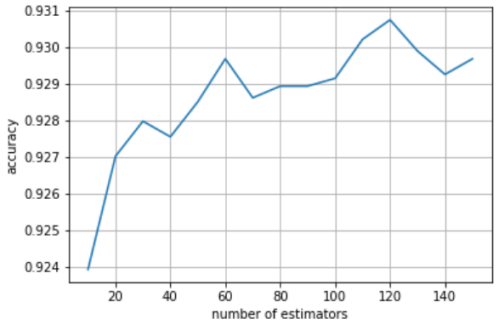

Figure 12: Figure showing how different numbers of estimators affect accuracy in LDA + RF model, for all colored images

From the plot above it is clear that even with a low number of estimators the accuracy is quite high. In order to avoid overfitting, the number of estimators chosen is 20. With this number of estimators the accuracy is found to be 0.9273. Moreover the time needed to perform LDA and then fit the model was 112 seconds, which is the lowest time for RGB images, when compared to the previous models.

Model Comparison

After having fitted three different models, there are some significant results. First of all it seems that for the CNN model for the grayscale images, not oversampling results in the testing accuracy being quite different than the training accuracy, something which suggests that the model may be suffering from overfitting. So this model was not very appropriate and for the rest of the project only balanced data were used. Another conclusion was that using colored images for CNN results in higher testing accuracy, when compared to using grayscale images. Moving on to the SVM model, when this was fitted with a sample of the grayscale images the accuracy was quite good, but the computational time was high. On the contrary when fitting SVM combined with LDA, it was possible to use the whole dataset of colored images. The accuracy was almost identical to the one of the SVM model with the sample of the grayscale images, but computational time was less. This suggests that the SVM + LDA model is more appropriate. Finally the Random Forest Classifier combined with LDA, was fitted to the whole set of colored images and gave a really good accuracy in a quite fast time. The results are summarised on the table below.

Model	Testing Accuracy	Time (seconds)
CNN (grayscale)	75%	178.7 s
CNN (RGB)	95%	383.88 s
SVM (grayscale/sample)	98%	673.24 s
SVM + LDA(RGB)	97%	578 s
RF + LDA(RGB)	92%	112 s

Figure 13: Table showing a summary of performance of the models chosen

From the table above, it seems like all models are quite good in prediction, but when taking into account accuracy and computational time the CNN or the FR+LDA for colored images may be the preferred ones.

Future Implications

Skin cancer is among the most common cancer types today. CDC mentions that skin cancer is the most common cancer type in the United States, and the American Cancer Society, mentions that the risk of getting melanoma "is about 2.6% (1 in 38) for whites, 0.1% (1 in 1,000) for Blacks, and 0.6% (1 in 167) for Hispanics". What is even more important, is the fact that skin cancer is highly treatable if found early. However if time passes and it remains undiagnosed then it is much more difficult to treat it [American Academy of Dermatology]. So being able to detect skin cancer and the type of it correclty is of great importance.

For this reason, having models that can identify the type of cancer which are fast and effective, is key. This project identified two more models other than CNN, which seem to work well for the dataset provided. However, for future work, these models could be trained with more data, where each class of cancer would be represented equally, in order to get more accurate results. Moreover, other models could be tested in order to see if they could provide better results and the existing models could be tested using better and faster software because the computational time could be faster, and more data could be handled. Finally, more testing examples could be provided in order to test the performance of the models even further.

Personal Reflection

In general, I found this project to be quite interesting but at the same time challenging. It was very interesting because introduced me to machine learning algorithms and models, for the first time. I really liked this, because I can see the direct application that a machine learning model can have on the real world. So, I believe that having even a very basic knowledge of machine learning is really helpful and it is a foundation which can be developed a lot if I would like to focus on this field. Also, it was very interesting that I got the opportunity to learn more than one models, because I got to understand how different algorithms work and how they can be combined. Moreover, it is now clearer to me how image classification works, how grayscale and colored images are handled, as well as how the data need to be shaped, in order to work with each model. Moreover, I learned how a combination of algorithms can lead to a really good result, where not only the accuracy of the model will be good, but also the computational time needed will not be that high. Finally, for this project I had to do a lot of research about the different models, how they work, and how they can be combined. With this I definitely had the opportunity to develop my research skills.

Regarding the challenges for the project, the first one I faced was the fact that it was a bit difficult to understand the initial data for the project and how these can be inputted in the models. This was solved after carefully examining the data and understanding the differences between the data files. Another challenge that I faced was my sense that the module introduced so many models, so I felt that I got a basic understanding of a lot of models, instead of a deeper understanding of less. That was difficult, because I had to do a lot of research to understand how the models work, and I still

feel that I have a lot of questions on that topic, even for models that I used. So, I think that, for me, would be more helpful to have analysed less models in depth. This would have helped me more for the project. Finally, I found difficult the fact that the tutorials for the module were a month before the project. So I did not have the opportunity to ask the questions that I wanted to ask, because my progress on the project was not that significant, back then. When it comes to future projects on the field, I think that what I would do differently will be to try and focus only on 2-3 models. In the beginning of the project, I was trying to choose between models and I spent a lot of time fitting some of the models introduced in the course, while if I had focused on less models, I could have done even more research for the models that I ended up choosing. For future projects, I would also try to stress a bit less, because in the end everything worked out and stressing less would have made the process more enjoyable at times. What I would keep the same in my future work, will be to have a structured plan like I had now and also start the project early in order to have enough time to understand the project and do any relevant research, just like I did.

.

.

.

.

.

.

Bibliography

Karthika, N., Janet, B. and Shukla, H., 2019. A Novel Deep Neural Network Model for Image Classification. International Journal of Engineering and Advanced Technology, 8(6), pp.3241-3249.

Shen, J., Tao, C., Qi, J. and Wang, H., 2021. Semi-Supervised Convolutional Long Short-Term Memory Neural Networks for Time Series Land Cover Classification. Remote Sensing, 13(17), p.3504.

Ballard, N., 2020. How to Resize Images Using Python. [online] Medium. Available at: https://towardsdatascience.com/how-to-resize-images-using-python-8aaba74602ed (https://towardsdatascience.com/how-to-resize-images-using-python-8aaba74602ed).

Hojman, E. and Conrad, A., 2016. Display image as grayscale using matplotlib. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/3823752/display-image-as-grayscale-using-matplotlib (https://stackoverflow.com/questions/3823752/display-image-as-grayscale-using-matplotlib).

Cerliani, M., 2020. How to feed into LSTM with 4 dimensional input?. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/47410239/how-to-feed-into-lstm-with-4-dimensional-input (https://stackoverflow.com/questions/47410239/how-to-feed-into-lstm-with-4-dimensional-input).

Carremans, B., 2018. Classify butterfly images with deep learning in Keras. [online] Medium. Available at: https://towardsdatascience.com/classify-butterfly-images-with-deep-learning-in-keras-b3101fe0f98 (https://towardsdatascience.com/classify-butterfly-images-with-deep-learning-in-keras-b3101fe0f98).

Chollet, F., 2016. Building powerful image classification models using very little data. [online] Blog.keras.io. Available at: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html (https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html).

Bhattacharrya, S., 2019. Support Vector Machine: MNIST Digit Classification with Python; Including my Hand Written Digits. [online] Medium. Available at: https://towardsdatascience.com/support-vector-machine-mnist-digit-classification-with-python-including-my-hand-written-digits-83d6eca7004a (https://towardsdatascience.com/support-vector-machine-mnist-digit-classification-with-python-including-my-hand-written-digits-83d6eca7004a).

Dominguez Garcia, C., 2021. Visualizing the effect of hyperparameters on Support Vector Machines. [online] Medium. Available at: https://towardsdatascience.com/visualizing-the-effect-of-hyperparameters-on-support-vector-machines-b9eef6f7357b (https://towardsdatascience.com/visualizing-the-effect-of-hyperparameters-on-support-vector-machines-b9eef6f7357b).

Malik, U., n.d. Linear Discriminant Analysis (LDA) in Python with Scikit-Learn. [online] Stack Abuse. Available at: https://stackabuse.com/implementing-lda-in-python-with-scikit-learn/ (https://stackabuse.com/implementing-lda-in-python-with-scikit-learn/).

GeeksforGeeks. 2021. SVM Hyperparameter Tuning using GridSearchCV | ML - GeeksforGeeks. [online] Available at: https://www.geeksforgeeks.org/svm-hyperparameter-tuning-using-gridsearchcv-ml/ (https://www.geeksforgeeks.org/svm-hyperparameter-tuning-using-gridsearchcv-ml/).

Vulovic, V., 2018. sklearn.metrics.accuracy_score vs. LogisticRegression().score?. [online] Cross Validated. Available at: https://stats.stackexchange.com/questions/354709/sklearn-metrics-accuracy-score-vs-logisticregression-score (https://stats.stackexchange.com/questions/354709/sklearn-metrics-accuracy-score-vs-logisticregression-score) .

Lopes, M., 2018. Is LDA a dimensionality reduction technique or a classifier algorithm?. [online] Medium. Available at: https://towardsdatascience.com/is-lda-a-dimensionality-reduction-technique-or-a-classifier-algorithm-eeed4de9953a (https://towardsdatascience.com/is-lda-a-dimensionality-reduction-technique-or-a-classifier-algorithm-eeed4de9953a) .

Nelson, D., 2020. Dimensionality Reduction in Python with Scikit-Learn. [online] Stack Abuse. Available at: https://stackabuse.com/dimensionality-reduction-in-python-with-scikit-learn/ (https://stackabuse.com/dimensionality-reduction-in-python-with-scikit-learn/) .

Thambawita, V., Strümke, I., A. Hicks, S., Halvorsen, P., Parasa, S. and A. Riegler, M., 2021. Impact of Image Resolution on Deep Learning Performance in Endoscopy Image Classification: An Experimental Study Using a Large Dataset of Endoscopic Images. [online] Available at: <https://doi.org/10.3390/ (https://doi.org/10.3390/) diagnostics11122183>

Mehri, A., 2017. Face Recognition Using Random Forest Classifiers Based on PCA, LDA and LBP Features. Master Thesis. Eastern Mediterranean University/Gazimağusa, North Cyprus.

Chen, R., Dewi, C., Huang, S. and Caraka, R., 2020. Selecting critical features for data classification based on machine learning methods. Journal of Big Data, 7(1).

Hasan, H., Shafri, H. and Habshi, M., 2019. A Comparison Between Support Vector Machine (SVM) and Convolutional Neural Network (CNN) Models For Hyperspectral Image Classification. In: Sustainable Civil and Construction Engineering Conference. IOP Conf. Series: Earth and Environmental Science 357 012035.

Xiong, T. and Cherkassky, V., 2005. A combined SVM and LDA approach for classification. In: IEEE International Joint Conference on Neural Networks. IEEE. pp. 1455-1459 vol. 3, doi: 10.1109/IJCNN.2005.1556089.

Chen, F. and Li, F., 2010. Combination of feature selection approaches with SVM in credit scoring. Expert Systems with Applications, 37(7), pp.4902-4909.

Don, R. and Prasanna, D. W., 2018. Multiclass Classification Using Support Vector Machines. Electronic Theses and Dissertations. Georgia South-eastern University.1845. https://digitalcommons.georgiasouthern.edu/etd/1845 (https://digitalcommons.georgiasouthern.edu/etd/1845)

.