PFEE Final Report

Majurca

Abstract This is the final report of our progress on the task of image classification for plastic waste sorting imparted to us by the company Majurca. Seen as this is the final report it will cover everything that has been done since the beginning of the project. This will be done in a chronological manner. As for an overview of what performed has been performed since the beginning of the project; For the image classification task we first chose a pretrained CNN model VGG16 for transfer learning. Then as our first data set was a subset of seven categories each containing 100 images we quickly faced issues regarding overfitting. To combat this employed regularization by adding an additional dropout layer to the VGG16 model. This allowed for a significant increase in validation accuracy (further detailed in subsequent sections). At this point we gained access to the complete dataset of approximately 64000 images. As we tested the previous model on the complete dataset we observed a significant disparity in the precision achieved on the initial dataset and the complete one. Upon further inspection of the data we concluded that this was due to the higher prevalence of images taken from different angles and with different luminosity on the complete dataset than in the initial one. After having informed the company of this change, they notified us that the complete dataset contained outdated images, and showed us some of the up to date ones. Upon further discussion with the company we found that, due to lack of labeling, there was no real way to query the dataset for the up to date images. So in order to retrieve only the up to date images from the dataset we instead implemented an image search engine. Finally, after using the correct images for the image classification task we achieved a final accuracy of 92%.

Jose A. Henriquez Roa
Christopher Diamana Lutete

January 26, 2022

Contents

1. Introduction

This section shall present the company Majurca as well as the task that was imparted to us from this one.

1.1. Majurca

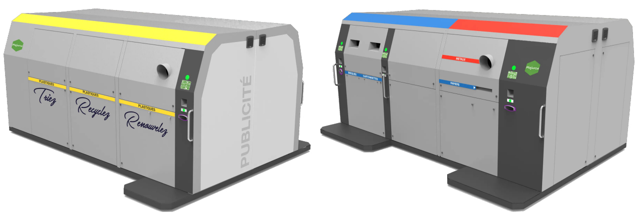

Figure 1: Plastic recycling module (Left). Metal recycling modules (Right)

The majurca company was founded on the 8th of October in 2013 by Stephanie Loss Letienne and as of 2019 there are currently two managing directors Benjamin Letienne and Valerie Paris Letienne. The company itself is listed in the sector of activities of specialized machine manufacturing. The main project that is currently being undertaken by the company is the one of creating a set of modular terminals for automatic classification and grinding of plastic waste to facilitate recycling. Both plastic and metal recycling modules are show in figure 1.

Both machines take the plastic or metal waste to recycle from the circular opening at the top right. Then depending on the type plastic or metal, they will first grind the waste and sort them into respective containers. When full, these container will then be taken out and transported to recycling centers. The two main points advertised by the Majurca company with respect this new method of recycling are:

1. The fact that the modules automate waste sorting on the more granular level actually required before grinding the plastic (e.i by the type of plastic or metal).

2. And the fact that by grinding the waste on site allows to directly export the ground waste directly to recycling centers.

Both machines still seem to be on the early stages of development with the plastic recycling one being further along the development process. Out of the two the one our group was involved with was exclusively the plastic recycling one. Our task was linked to the plastic type recognition step at the very start of the pipeline.

1.2. The task

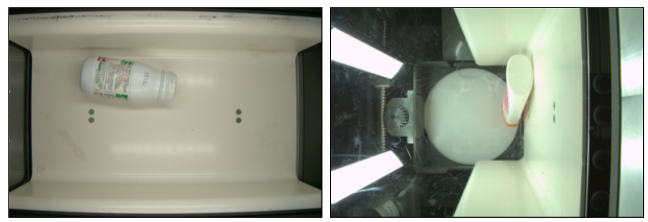

Figure 2: Top angle (Left). Side angle (Right)

The mechanism that is currently being considered for the task of classification of the input waste is image classification through deep learning models. Our task was to implement such a model.

The images are taken through the following process; First the plastic waste is deposited into a chamber that lies right after the circular opening at the top right of the machines. Two images are then taken for each deposited component from different angles. One from the top and one from the side. Two examples of this can be found on figure 2. Our task would then be to train image classification models to detect the type of plastic of the objects inserted in the chamber. For this we could either train a single model to detect the plastic type despite the angle or we could train two separate models. One for each image angle. This is further discussed in the following sections.

This was the extent of our task given to us by Majurca. It was a simple image classification task. The classification part itself fairly straightforward. Were we found most of the complications while working on the project was during the communication with the actual company. These issues will be detailed in the following sections. However, we do note that everything was eventually sorted out.

2. Project organization

This section shall present everything related to the our communication with the Majurca company as well as the task repartition of each of the member of our group. First we shall present the Majurca representative that was in charge of communication with us. We shall then present the communication tools that were used during our interactions. After that shall detail the interaction stages we had with the company. And finally, give a presentation of each of the members of our group as well as the tasks we were individually charged with.

2.1. Our interlocutors

All through the project we have had multiple interlocutor on the EPITA side, and a single one on the company side.

The company representative that supervised our work all throughout the project was Stephanie Loss Letienne. As previously mentioned, she is the founder of the Majurca company. She took charge of both assigning tasks and supervising our progress.

On the epitome side, we have had three interlocutors, chronologically these were:

1. Pierre-Julien Grizel was the first to direct our interaction with the company. He was present during the first few meetings with this one. In these meetings he helped us both us and the company with a few technical details mostly regarding how to get access to the image dataset. During these meetings he also helped answer some of Stephanie's questions regarding Deep Learning models for image classification, such as power consumption and inference time.

2. Cedric Joly was our second interlocutor on the EPITA side. By the time the change was made we had been in charge of most of the interaction with the company for a while. Pierre-Julien Grizel would however still help with things related to the image dataset. Mostly by given us directions on how to access this one through email. Right after the change we were instructed that we would be required to do monthly reports through teams meetings of everything that had occurred during the month with the company. After having updated Cedric Joly on everything that had happened with the company from the the start of the project he would give some additional advice on new methods we could improve our accuracy in the image classification task. These meetings lasted for about three months.

3. Nicolas Boutry was our last interlocutor. The change in interlocutor took place right before the last meeting with Cedric Joly had been scheduled (which was also to be the very last one before the final presentation) and so most of the interactions with him were through email.

2.2. Communication tools

Three tools were used for communication between the company and the EPITA interlocutors.

These were; Teams, email and the Whereby.com website for video calls. Following we shall give a description of how these communication tools were used and with whom.

1. *Whereby.com* This was the first tool we used to communicate with Stephanie Loss Letienne and Pierre-Julien Grizel. This is where first few meetings with Pierre-Julien Grizel took place as well as all of the subsequent ones with Stephanie Loss Letienne.

   Whereby is a video call website that advertises much greater ease-of-use when compared to some of the other video call applications. With Whereby, meetings are held directly through their website. It is only required for clients to reserve a URL, which is then meant to be shared with all of the members that will be attending the meeting.

2. *Email* Most communications regarding the video call meetings times as well as communications regarding some of the technical details related to the image classifications (e.g dataset related issues) tasks were done through Emails.

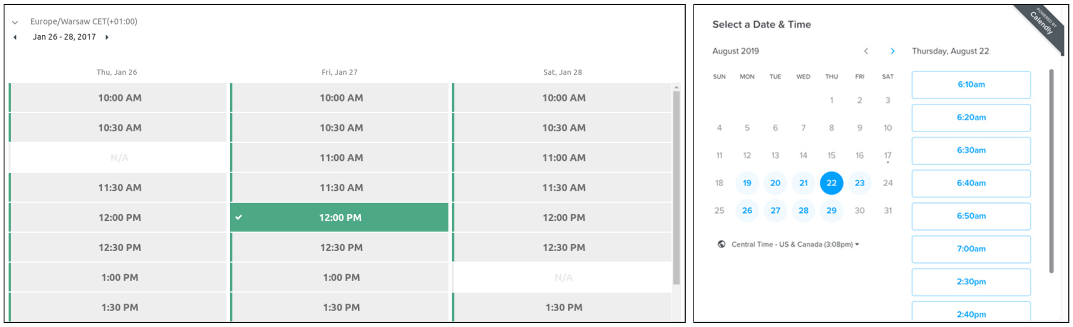

Figure 3: Appoint.ly (Left), Calendly (Right)

3. *Teams* This application was exclusively used for the monthly reports with Cedric Joly.

Two additional tools were used for coordinating the meetings dates and times. These were Appoint.ly and Calendly.

1. *Appoint.ly* This application was used to coordinate the first few meetings with Pierre-Julien Grizel and Stephanie Loss Letienne. A screenshot of the interface of this application can be found on figure 3 on the left hand side.

2. *Calendly* This one was used for coordinating the Teams meetings with Cedric Joly. A screenshot of the interface of this application can be found on figure 3 on the right hand side.

2.3. Interaction Stages

For our interaction with the majurca company we count four main stages. These were separated by the moments in which we got access to different portions of the image dataset.

1. *Task definition & initial dataset* This stage starts with the first meeting. In this one we were given further details on what the mission would be. For instance we were showed for the first time some example images which we would be using for the classification task all throughout the project.

   We were then informed as to where the classification model would be used, this was inside the recycling modules. We were then showed how the images were taken inside the machine and what type of variances we could expect from one image to another, so that we could take this into account during training.

   After this first meeting we were given a small subset of the complete dataset consisting of 700 images divided in between seven categories of 100 images each.

   This concluded the first meeting, and so the first stage.

2. *Initial dataset progress report* After getting access to the initial dataset we were also given some time to test out some initial models for classification. This stage spanned through the following three meetings after the initial one, which was about a month worth time. During each of these meetings we would report each of the classification models we had tested out. We would also highlight some of the regularization methods we had employed to deal with the small dataset. At the end of each meeting we would state which of the models we had tested had performed best for the given dataset and would highlight some of the performance metrics that we used to determine this (e.g accuracy, confusion matrix etc.).

3. *Complete Dataset* After having produced some initial results we were given access to the complete dataset. This one one was composed of a much higher number of images as well as a much higher number of categories. There were however some issues in the quality of the new images in the dataset. This will be further discussed in later sections. After having gotten access to the complete dataset we entered to

following final stage.

4. *Continuous progress report* Upon getting access to the full dataset all that was left was to keep training and modifying models iteratively in order to get the highest accuracy we could from the data. This stage continued into the very last meeting. During every meeting we would present the improvements over the accuracy we had achieved over the last versions. As well present the methods we had employed to improve the latest models (e.g data augmentation, regularization, changes to the ANNs architecture etc.).

2.4. Individual task repartition

Our group started with three people Félix Auneau, Christopher Diamana Lutete and Jose A. Henriquez Roa. However, soon after the second meeting meeting with Majurca Félix Auneau seemed to have left the school. And so this section shall only present the work repartition between Christopher Diamana Lutete and Jose A. Henriquez Roa. The following sections shall be presented by their respective individual:

- Jose A. Henriquez Roa: During the first semester of the project Félix Auneau and I were the ones that contributed to the training and testing of the classification models. Félix Auneau did most of the work for the first meeting and I took charge for the work of the following meetings after he left. Then to better balance out the work distribution in between the remaining two members for the second semester, a decision was made to alternate the work that had to be done before each meeting in between Christopher Diamana Lutete and I.

- Christopher Diamana Lutete: During the second semester of the project, I tried to ameliorate the classification models. Also, I cleaned the dataset by detecting the wrongly labeled images.

3. The assignment

This section shall present a detailed description of everything that was done for the image classification task given to us by Majurca. It shall first present the task as defined by the company. And then the detailed description of our work. The following section shall then present a conclusion to this report by first present the final results as well as of the possible remaining improvements that could still be implemented given some additional time and resources.

3.1. Task assignment

During the first meeting the task given to use was fairly straightforward. We would be required to implement an image classification model that, when given an photo of a single plastic object, would be required to determine the type of plastic of said object. For the different categories of types of plastics we were given seven in total. There were no issues or misunderstandings as to what the task would be.

3.2. The work

This section shall chronological present everything that was done to complete the task. Each new subsection will present the methods that were used to improve upon the issues encountered in the preceding one.

3.2.1. Hand made model architecture

Upon receiving the initial dataset of 700 images we first decided to try and see what kind of results we would get by using a hand made model. In more time constrained scenarios it would have been more logical to go directly to transfer learning. However, since we knew the project would be lasting about a year we wanted to be as exhaustive as possible from the start.

The model itself was composed of two convolution layers and one fully connected layer. It total there were `1,699,399` trainable parameters. After trying out a considerable amount of variants this one was the one we found performed the best for the training set. We tried to keep the amount of trainable parameters as small as possible and the architecture as shallow as possible in order to combat overfitting as much as we could. A full description of the model can be found on figure 11.

However, seen as the initial dataset was fairly small and by the fact that manually finding an architecture that does not overfit is fairly rare and time consuming. We did not end up finding any acceptable results on the test set of images. On average we found 8% accuracy on this one.

And so we decided so after to discard the idea of making a hand made model in hopes of finding better results through transfer learning.

3.2.2. Transfer Learning

Transfer Learning refers to the act of using other Artificial Neural Networks (ANN) that have been trained for separate but similar tasks. And incrementally modifying the learned parameters by retraining the network on a new dataset in order to make it useful for a different task than the original. The most popular networks that are chosen for transfer learning are the ones that are known to both achieve high results on similar tasks than the one being tackled and that are able to generalize what they have learned to other never seen tasks.

In the case of computer vision, pretrained neural networks are usually judged on their performance on the ImageNet dataset. This is due to the fact that the ImageNet dataset is composed of over 18 million images with more than 20 thousand categories such of things are have very little in common with each other, such as “balloons”, “strawberries”, “dog” and more. If a network performs well in such a task it is almost sure to be able to generalize to other ones fairly well.

For our task we tested out all of the best performing networks on the ImageNet task, a list of all of the models we tested can be found on `https://keras.io/api/applications/`, here you shall find some information on the architecture of each model as well as their respective accuracy on ImageNet.

Out of all the models we tested, the ones we found to be the most promising for the current task were VGG16, InceptionV3, ResNet50, EfficientNetB0 and Xception. And out of these ones the one we ended up choosing was VGG16. Despite not being the best performing one on ImageNet, VGG16 is relatively small, which means it is the easiest one to train out of the ones previously considered. VGG16 is also very widely known to be able to generalize very well on a relatively small number of epochs than other models. This was also one of the contributing factors that lead us to make this choice. It is however noteworthy to state that some of the other networks are still likely to outperform VGG16 for the current task if given a much higher number of epoch of training. We made the choice to use VGG16 mostly due to the lack of computational resources.

Aside from the pooling layers VGG16 is composed of 16 layers, 13 convolutional layers and 3 fully connected layers. With a total of `138, 357, 544` trainable parameters. The full details on the architecture can be found on figure 12. With no additional modification to the model we achieved an accuracy of 75% on the validation set, which is a significant improvement over the last one. The following shows the learning curve of this particular model over 300 epochs:

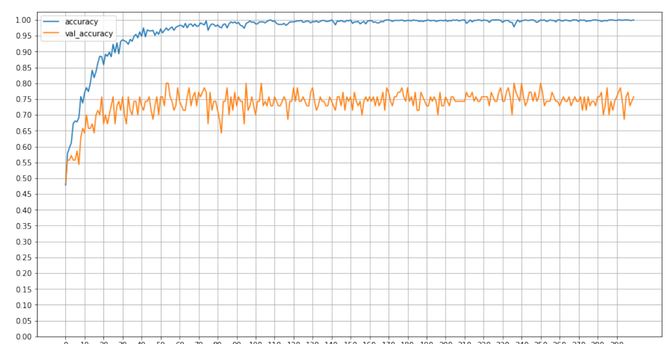

Figure 4: VGG16 Learning curve

As can be seen in figure 4 there is still a high amount of separation in between the validation accuracy and the test accuracy. Which means that the model is still overfitting the training data. To help with this we applied some regularization; as the following section will present.

3.2.3. Regularization

Applying regularization to an ANN model refers to the action of constraining the weights that the network is trying to learn during training. This can be done through a number of different methods all of which have varying results. The ones most commonly used for deep neural networks are L1 regularization, L2 regularization and Dropout. Depending on the method that is used, constraining the weights can have different effects.

Some regularization help with the vanishing and exploding gradients issue commonly seen in neural networks. The vanishing and exploding gradients issue refers to the fact that depending on the input data the computed gradients used to update the weights of the network can either be vanishing, meaning they will have almost no impact when updating the network, and so it will be as if the data was never used. Or the gradients can be exploding, meaning that the weights will tend to complete overwrite any previous value already in the during, hence forcing the network to forget about any previous data, this also referred to as catastrophic interference or catastrophic forgetting. The methods that are known to help with this particular issue are L1 and L2 regularization as well as another method called Batch Normalization, this later one however is not just a regularization method but helps with this non the less.

With model implemented on the last section there was no vanishing or exploding gradients gradient issue. There was however significant overfitting. A model that overfits is one that performs well on the training set and relatively poorly on the validation set. In other words it is memorizing the training data and by doing so it is not able to generalize properly to data that it has not seen. Regularization can also help with this. One of the regularization method that helps the most with overfitting is dropout. Dropout gets its name from the fact that any network that implements it will have some of their nodes its nodes disabled or dropped out during training by a predefined probability. The way that dropout helps with overfitting is somewhat hard to explain. However, it can be interpreted that dropout obstructs a networks' tendency to memorize the training data by forcing said data to be learned by multiple random nodes distributed all throughout the network. This in turn hinders to possibility of multiple neighboring nodes being codependent and memorizing the same portion of the input data all together.

It is noteworthy to state however that regularization methods will often increase the time required for a machine learning model to converge to a good solution. This is specially true in the case of dropout. This in turn translates to the increased need for more computational resources or time to train the models.

Going back to the topic of our model. After testing out many different dropout layer configurations on the initial architecture. We found the best results by simply substituting one of the fully connected layers by a dropout layers on the VGG16 model. Some additional modifications were made to the models in regards to the fully connected layers. We found that to counteract the increase in training time induced by the dropout layer it would be best to decrease the number of nodes of the fully connected layers of the VGG16 model. After these modifications to the model we decreased the number of trainable parameters from `138,357,544` to `14,847,815`, the full architecture can be found on figure 13. And the accuracy was increased from 75% to 88%. The following plot shows a comparison in between the learning curves of the initial model and the new one:

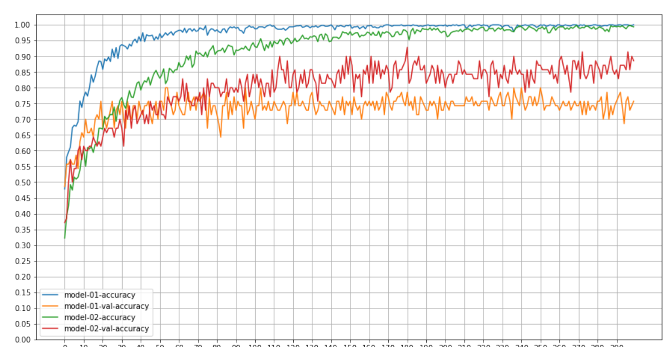

Figure 5: VGG16 and VGG16 with dropout comparison. Train accuracy and validation accuracy of the initial model are respectively the blue and orange curves. Train accuracy and validation accuracy of the new model are respectively the green and red curves.

As shown in the above graph, the validation accuracy of the new model quickly surpasses that of the previous model during training. However, even with the considerable decrease in the number of trainable parameters, the model still takes much longer than the initial one to converge to a solution. Again, this is due to the dropout layer.

3.2.4. Content-based image retrieval

From this point on most of the improvements on the accuracy were attained by making changes to the data being feed to the network.

As previously mentioned we initially had access to a dataset composed of about 700 images. We were later then given access to a dataset of about 64000 images, this was during the regularization stage. After training the model on the new dataset we had noticed a slight decrease in the accuracy of the model when compared to the one achieved on the initial dataset. Initially we had though that this was due to the possibility that the samples from the initial dataset were very similar to each other and so made it easier to generalize what was learned on the training set to the validation set. However, after looking at the new data we had noticed that there were some slight differences in the images between the old and the new dataset.

As previously mentioned the modules were meant to take images from two angles, one from the side and one horizontally. Within the new dataset we found that these images showed some slight differences:

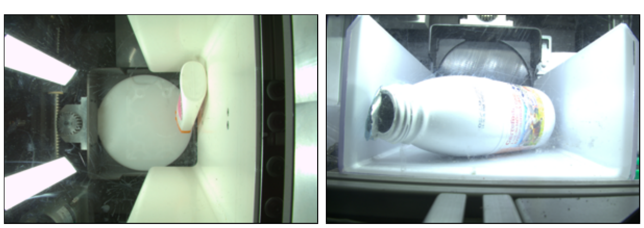

Figure 6: Image taken from the side.

For images taken from the side we noticed that some were taken horizontally and others vertically. For some images that we taken from the side we also notices some slight differences with the tint of the images. For images taken from the side some had a more greenish tint than others. This is slightly present in the leftmost image of figure 6.

Figure 7: Image taken from the top.

For images taken from the top the differences we more subtle. In figure 7 we see that there were some slight changes in the positioning of the camera. We also found some additional differences in regard to the tint. In which some images taken from the top had a more yellowish tint than others.

Upon asking some questions regarding this issues we were told that the dataset contained images from both old machines, which had different camera angles at the time, as well new new ones, with more up to date angles and lighting. However, looking at the SQL file we had used to download the complete dataset we found that there was no tag indicating from which machine each picture from. We only had examples of what the up to date images looked like. And seen as we did not have the time to go through the entire dataset in search for similar ones, we decided to implement a Content-based image retrieval to search for the similar images in the dataset.

For this we used a python library called DeepImageSearch which implemented both the indexing and the search logic when given an input image dataset. In the figures 9 and 8 the two examples we used for the query are the leftmost ones and all of the others are there search results.

The results we got after the search was a set of image indexes pointing to the images most similar to the query images all sorted by similarity. Since the query images were from the second machine all that was left was to go through the images and manually find the cutoff points in which the images started being from the old machine. After doing this for the seven categories we ended with a dataset that made

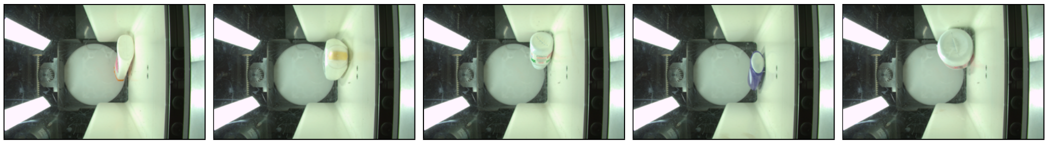

Figure 8: Side angle content-based image retrieval

Figure 9: Top angle content-based image retrieval

up of only the images from the latest machine.

After removing the old images from the dataset we were left with a new one made up of about 53 thousand images. By retraining the model on the new dataset we got a validation accuracy of 92%. With the following learning curve:

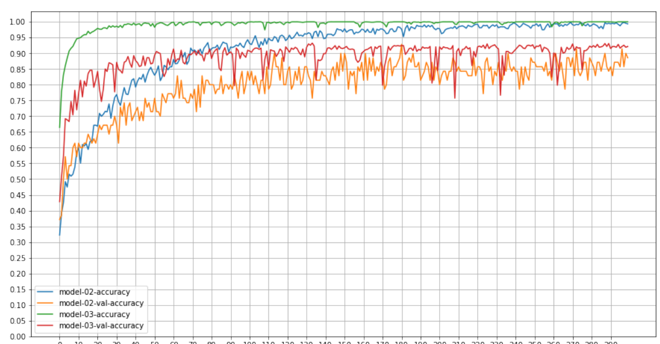

Figure 10: VGG16 with dropout comparison with the old and new dataset. Train accuracy and validation accuracy of the model trained on the old dataset are respectively the blue and orange curves. Train accuracy and validation accuracy of the model trained on the new dataset are respectively the green and red curves.

As shown in figure 10, removing the image of the old machine greatly increased convergence time. This may be due to the fact that despite being images of the same objects but with slightly different angles, the old images must have been acting as noise more than actually helping with the classification task.

4. Conclusion

4.1. Results

By making use of Transfer Learning and Dropout for Regularization and by extracting the relevant images from the provided dataset through Content-based image retrieval we were able to manage an 92% accuracy on the given image classification task.

4.2. Learnings

Through this project we were able to apply everything that we learned during the courses into a two semester long image classification task. Through this one we made use of all the theory given to us during the courses to tackle a real world task. For instance we were able to get real world practice on the topics of Computer Vision and Content-based image retrieval we had respectively learned during our Introduction to Neural Network and Machine Learning for Pattern Recognition classes.

4.3. Possible Improvements

Aside from being limited in terms of time we could dedicate to this project due to the other university projects and classes, we were also somewhat limited in terms of compute resources. Seen as we only had access to our local machines for this project we were mostly limited by our own personal hardware. This in turn translated to an inability for us to test out some of the more advanced and complex models. Hence why we kept VGG16. And so we believe if we had been able to get access to cloud services it could have improved our changes at getting a higher accuracy.

The last thing that would have served as an improvement would have been for us to ask more questions regarding the dataset from the very beginning of the project. It was only fairly late during the project that we ended finding the the dataset had a substantial amount of out-of-date picture in it. If we had found this out sooner we would have been able to train the initial models of the correct data. Which in turn could have led us to make different changes in the architecture that in turn would have achieved a better final accuracy.

5. Annex

5.1. Model Architectures

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 200, 300, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 100, 150, 32)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 100, 150, 64)      18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 50, 75, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 240000)            0
_________________________________________________________________
dense (Dense)                (None, 7)                 1680007
=================================================================
Total params: 1,699,399
Trainable params: 1,699,399
Non-trainable params: 0
_________________________________________________________________
```

Figure 11: Hand made model

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

 flatten (Flatten)           (None, 25088)             0

 fc1 (Dense)                 (None, 4096)              102764544

 fc2 (Dense)                 (None, 4096)              16781312

 predictions (Dense)         (None, 1000)              4097000

=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0
_________________________________________________________________
```

Figure 12: VGG16 model

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, None, None, 3)]   0

 block1_conv1 (Conv2D)       (None, None, None, 64)    1792

 block1_conv2 (Conv2D)       (None, None, None, 64)    36928

 block1_pool (MaxPooling2D)  (None, None, None, 64)    0

 block2_conv1 (Conv2D)       (None, None, None, 128)   73856

 block2_conv2 (Conv2D)       (None, None, None, 128)   147584

 block2_pool (MaxPooling2D)  (None, None, None, 128)   0

 block3_conv1 (Conv2D)       (None, None, None, 256)   295168

 block3_conv2 (Conv2D)       (None, None, None, 256)   590080

 block3_conv3 (Conv2D)       (None, None, None, 256)   590080

 block3_pool (MaxPooling2D)  (None, None, None, 256)   0

 block4_conv1 (Conv2D)       (None, None, None, 512)   1180160

 block4_conv2 (Conv2D)       (None, None, None, 512)   2359808

 block4_conv3 (Conv2D)       (None, None, None, 512)   2359808

 block4_pool (MaxPooling2D)  (None, None, None, 512)   0

 block5_conv1 (Conv2D)       (None, None, None, 512)   2359808

 block5_conv2 (Conv2D)       (None, None, None, 512)   2359808

 block5_conv3 (Conv2D)       (None, None, None, 512)   2359808

 global_average_pooling2d (G  (None, 512)              0
 lobalAveragePooling2D)

 dense (Dense)               (None, 256)               131328

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 7)                 1799

=================================================================
Total params: 14,847,815
Trainable params: 14,847,815
Non-trainable params: 0
_________________________________________________________________
```

Figure 13: VGG16 model with dropout