* To summarize your course project in a video of 10 - 15 minutes long
* Upload to Youtube and submit link
* Some examples for reference:
  + <https://www.youtube.com/channel/UCSBrGGR7JOiSyzl60OGdKYQ>
  + <https://www.youtube.com/channel/UC_sfvZvvPUbOQhDs_cqlx_A>

| Content | Script | Reference Duration (seconds) | Who |
| --- | --- | --- | --- |
| Introduction | Hello everyone, we are group 12. The project we have chosen to work on is a kaggle competition, Plant Seedlings classification. The group consists of Aaron, Benedict, Ray and Guan Wei.  NEXT SLIDE  The agenda for this presentation is as follows: first, we will talk about the problem statement. Then, we will talk about exploratory analysis, data preprocessing, the methodologies used, solution novelty and challenges. Lastly, we will conclude the presentation.  NEXT SLIDE | 20-25s | GW |
| Problem | Starting off with the problem statement, the presence of weeds amongst growing crops poses a significant challenge for the agricultural industry.  PRESS    These weeds compete with crops for resources, which results in poorly grown crops.  PRESS  If these weeds are removed late, it will risk the crops being uprooted as mature weeds will get entangled with the crops.  NEXT SLIDE  Looking at these images, are you able to tell which is a weed?  <image >  They may look similar, but the one on the right is a weed.  NEXT SLIDE  Differentiating weeds from crops is no easy task, but it is an essential one. In large farmlands, it is not practical to manually check for and remove weeds. Thus, this task is often assigned to robots. As such, there is a need to train models capable of identifying the various categories of crop seedlings.  NEXT SLIDE  The aim of the project is to determine an optimal learning strategy for classifying the different categories of plant seedlings. The ultimate goal being that the model will be able to detect and categorise the crop and weed seedlings to a high level of accuracy.  NEXT SLIDE | 55s | GW |
| Exploratory Data Analysis | Moving on to exploratory data analysis.  An overview of the data provided by kaggle tells us that the dataset consists of a set of labelled images for training and a set of unlabelled images for testing. This table shows the classes of plant species in the training dataset. There are a total of 4750 images across 12 categories. While there are a total of 794 images in the test dataset.  NEXT SLIDE  Looking at the data distribution of the train dataset, we can observe that different categories have a different distribution, it is non-uniform. The category with the highest image count, has almost 3 times the count of the category with the lowest image count.  NEXT SLIDE  Looking at the size distribution, we can see that they are of varying sizes. In this figure, we can see that in the train and test datasets, the image sizes vary widely.  NEXT SLIDE  And, in this figure, we can see that even amongst categories, the sizes of the images vary as well.  NEXT SLIDE  Lastly, looking at the histogram properties of the dataset. Analysis on the colour histograms of the dataset gives us insights on the colour distribution across the categories. It can be observed that the intensity levels are very different across the categories.  NEXT SLIDE | 60s | GW |
| Data preprocessing | Moving on to Data Preprocessing. In the data preprocessing step, we did 4 things: data augmentation, image resizing, image masking and train test validation split.  In data augmentation, we performed 5 possible augmentations to the input images for each category.  NEXT SLIDE  These augmentations were chosen at random, and were repeated until each category had a total of 1000 images.  NEXT SLIDE  For image resizing, as our approaches required images to be of fixed input shape, we had to resize all images to the same dimensions. The final dimensions chosen were: 256x256, this ensured that sufficient details and features were retained in our images, but at the same time, we did not spend excessive resources to train our models.  NEXT SLIDE  For image masking, it was done as we wanted to remove undesired noise in the background, such as soil, or the barcode label. As the seedlings are mostly green in colour, the pixels that make up the seedling’s leaves and stems were masked out. These were the steps performed to mask the images: first, we used gaussian blur, then the image was converted to HSV format, then a mask was created and transformed to a boolean mask. To finish up masking, this mask was applied on the original image.  NEXT SLIDE  Lastly, for the train test validation split, the dataset was split into training and validation datasets in the ratio of 9:1.  NEXT SLIDE  Now, we will talk about our various methodologies and how we approached the problem. Ray will continue talking about it. | 90s | GW |
| k-NN | Thank you Guan wei. We will now be discussing different models that we have experimented with. We started off with some of the classic models.  NEXT SLIDE  The first model we tried out was the K-nearest-neighbour (KNN). KNN performs the classification by calculating the proximity of the data point and the neighbours. The number of neighbours is determined by the k-value. We implemented K-NN with default parameters and achieved 59.4% for Kaggle score. This result is used as a baseline for other models  NEXT SLIDE | 25s | Ray |
| SVM | The second model is the Support vector machine. SVM performs the classification by finding an optimal hyperplane which maximises the margin between different classes. Hyperplanes are drawn in such a way that data points of different classes are on different sides of the hyperplane. During the experiment, we used GridSearchCV to find ideal values for parameters, which are used to implement SVM. We achieved a Kaggle score of 69%.  In general, we noticed that classic models have poor performance. Therefore, we decided to explore a more efficient model which is Convolutional Neural Network. Now I will pass the time to Aaron to continue.  NEXT SLIDE | 45s | Ray |
| Xception | Thank you Ray. As Ray mentioned, we use convolutional neural networks, CNN, to improve the classification accuracy of the test images. Each of the CNNs we used were pre trained on over a million images from the ImageNet database.  NEXT SLIDE  The first CNN we used is Xception, which is an extreme form of the inception module concept. It replaces the inception modules with depthwise separable convolutional layers spanning both space and depth to high degrees. Many such convolutional and pooling layers run in parallel to each other, making the model less prone to overfitting, while reducing computation time through parallelism.  Additionally, Xception also applies the depth filters on the inputs prior to performing convolution operations on them. It does not perform any batch normalisation after each summation block nor use any non-linear activation functions after each convolutional layer. This improves the performance of Xception to varying degrees over other models.  The final accuracy of the Xception model is 96.095%.  NEXT SLIDE | 60 secs | Aaron |
| EfficientNet | The second CNN we used is EfficientNet. This model is doing a Neural Architecture Search (MAS) using AutoML MAS framework that optimises for both accuracy and efficiency (FLOPS) to reduce computation.  The resulting architecture uses mobile inverted bottleneck convolution (MBConv block) which is also used in MobileNetV2 and MnasNet. Hence, it is optimised for accuracy but penalised for if it is being too computational.  That said, this model achieves both higher accuracy and better efficiency over other existing CNNs, reducing parameter size and computational FLOPS. The final accuracy of the EfficientNetB7 model is 97.103%.  Now I will pass the time to Benedict to continue with the rest of the CNNs.  NEXT SLIDE | 30 secs | Aaron |
| InceptionResNetV2 | Thank you Aaron, the third CNN we used in this project is InceptionResNetV2. This model is a hybrid combination between the traditional layers of an Inception based CNN and the use of scaled residual connections between the layers of a ResNet based CNN. These connections replace some of the typical pooling layers of the CNN and bypass the neurons in some convolutional layers during training. This helps the model avoid the vanishing gradient problem, while also reducing training time. The subsequent use of activation scaling also helps to improve its training stability.  Transfer learning was similarly applied for InceptionResNetV2 using the same optimizer, loss type and hyperparameters as the Xception model. The final accuracy of the InceptionResNetV2 model is 97.103%.  NEXT SLIDE | 60 secs | Benedict |
| Ensemble Learning | After retraining the individual adapted CNNs used in this project, we used a majority voting ensemble learning strategy to improve the final accuracy score of the test images class predictions. The prediction vectors from each of the 3 CNNs for each image were summed together into a single vector of probabilities. The argmax function is then used to determine the index of the element in that resultant vector with the highest probability value. Hence, the class to which that index belongs to is considered as the final predicted class of the input image.  The final accuracy score for the majority voting ensemble of all 3 CNN models is 98.362%. I will now pass the time to Ray to begin on our project solution novelties.  NEXT SLIDE | 50 secs | Benedict |
| Transfer learning & fully connected layers | Thank you, Benedict. Firstly, we made use of existing pre-trained CNN models to save on time and computational resources. These models have transferrable learned features from previous training, which allow us for faster retraining for classifying the seedling images.  To prevent the influence of the model’s previous training, we decided to implement our own fully connected and output layers as our second novelty. Each fully connected layer used up to 2048 neurons to capture more features. We further customised these layers by adding dropouts which helped to prevent overfitting. Now i will pass my time to Aaron to continue  NEXT SLIDE | 42s | Ray |
| Ensemble learning & Early stopping callback | Next, we implemented ensemble learning with the 3 adapted CNNs to take advantage of the fact that two or more heads are better than one. Using a majority voting strategy across all 3 CNNs, our ensemble successfully achieved a final accuracy score higher than any of the individual CNNs.  The first callback we used is the early stopping callback, which stops the training of the model once a monitored metric fails to improve beyond a specified number of epochs, which is also known as the patience value. Stopping the training of a model at that point ensures that it does not overfit on the training data, while reducing the amount of computational time required by avoiding unnecessary epochs. The monitored metric used in all the CNN models was validation loss with a mode of minimum and a patience of 5.  I will now pass over my time to Benedict to continue on our solution novelties.  NEXT SLIDE | 60 secs | Aaron |
| Model checkpoint callback & learning rate scheduler | Thank you Aaron, the second callback we used is the model checkpoint callback, which saves the weights and biases of the model being trained to disk whenever a specified metric being monitored shows improvement. The metric we used for this callback is also validation loss with a mode of minimum, meaning that the model is saved to the disk every time its validation loss during training decreases to a new minimum value. Afterwhich, the best weights and biases of our models can be subsequently loaded from disk directly for making future predictions.  The next novel strategy employed is the gradual and linear decay of the learning rate by 5% in each epoch after the 5th epoch of training. While a larger learning rate used for the first five epochs helps to update the weights and biases faster, the gradual reduction in learning rate afterwards helps the model reach the local or global minima more effectively by taking smaller update steps. This callback was observed to improve the generalisation of our models and their final accuracy score.  I will now pass the time to Guan Wei to continue on the solution novelties.  NEXT SLIDE | 60 secs | Benedict |
| Dropouts & lvl 2 regularisation | Thank you Benedict.  In the project, even though the large number of neurons help to improve the results, it can also be prone to underfitting or overfitting. To address this problem, we introduced dropouts of 0.5 after each dense layer.  This means that in each epoch, 50% of the neurons will be disabled by setting their output to 0. This forces the remaining neurons to have their updates scaled up to learn more effectively, preventing the overfitting or underfitting of specific neurons.  Lastly, we used level 2 regularisation. The regulariser minimises the complexity of the model by penalising the square value of each neuron’s weight. This reduction in weight value by a small percentage ensures that they do not fluctuate significantly. Doing so ensures that the model does not learn overly complex features from the training data to reduce the likelihood of overfitting.  NEXT SLIDE | 55s | GW |
| Challenges | Now, moving on to the challenges. The main challenge we faced during this project was the limitations of computing resources. As we did not have enough computing resources on our personal computers, we turned to google colab and the kaggle kernel. However, the free resources were limited. Thus, we had to circumvent these limitations.  NEXT SLIDE  What we did was that we preprocessed and saved the images instead of doing the preprocessing during model training, NEXT SLIDE we also made sure our image size was not too big.  NEXT SLIDE  Then we made sure to save a checkpoint whenever results improved during training,  NEXT SLIDE  and lastly, we even created multiple accounts so that we could continue training when we run out of free resources on one account.  NEXT SLIDE | 30s |  |
| Conclusion | Finally, moving on to the conclusion. The approach that gave the best results was the ensemble learning approach. Thus, that approach was chosen and our final score comes in at 98.362% and a rank of 77.  NEXT SLIDE  Lastly, we conclude that many machine learning problems and their applications currently, are beginning to require and rely on near perfect accuracies in order to achieve their goals effectively. To achieve high prediction accuracies, similar to our project, it is likely that two or more convolutional or deep neural networks would need to be adequately trained and combined in ensemble learning.  This brings us to the end of our presentation. Thank you for listening. | 35-40s | GW |
| Total | |  |  |