Deakin Simpsons Challenge 2021

SAMUEL ANASTASSIADES, Deakin University, Australia
UDAY KUMAR GUMMADI, Deakin University, Australia
PUJIT YADALA CHENCHU, Deakin University, Australia

The Deakin Simpsons Challenge is a computer vision competition in which students attempt to use machine learning to recognize Simpsons characters individually in images. In this paper we use tensorflow to develop a Deep Convolution Neural Network to perform image recognition. For our base model we use InceptionV3, pre-trained on ImageNet. We apply fine-tuning techniques to improve the model fit. We also investigate the use of data augmentation as a method of data generation and discuss it's limitations. With this method were able to develop a model which achieves a 96.5% accuracy on final testing data.

ACM Reference Format:
Samuel Anastassiades, Uday Kumar Gummadi, and Pujit Yadala Chenchu. 2021. Deakin Simpsons Challenge 2021. In *Proceedings of Deakin Simpsons Challenge 2021 (Simpsons Challenge 21).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The Deakin Simpsons Challenge is a computer vision competition in which students attempt to use machine learning to recognize Simpsons characters individually in images. The topic of image recognition is well covered in the literature [4, 5] and it is widely accepted that deep convolution neural networks (CNNs) are the standard for image classification[4].

CNN's work by successively identifying image features in each layer. In this way they can identify and classify image categories with fewer parameters than a fully connected dense network. This is advantageous as it means CNNs are less prone to overfitting, and faster to train.

2 BUILDING THE MODEL

For this task we considered several factors including network architecture, hyper parameter tunning and what pre-processing the dataset would need. We began by examining the dataset, and performing pre-processing tasks, such as rescaling and data augmentation. We then experimented with various model architectures, before deciding to use a pretrained model. Finally, we optimized our model performance by tuning the relevant hyper parameters.

2.1 The Dataset

The dataset provided contains 20 categories and just 19,548 images, of which 10% would be needed as validation set, leaving 17503 images for training. In general, more data leads to a more robust model, which is less prone to overfitting. While the competition allowed for additional data to be gathered from external sources, we found that simply incorporating a data augmentation pipeline into our model allowed us to train a robust model without extreme overfitting.

Manuscript submitted to ACM

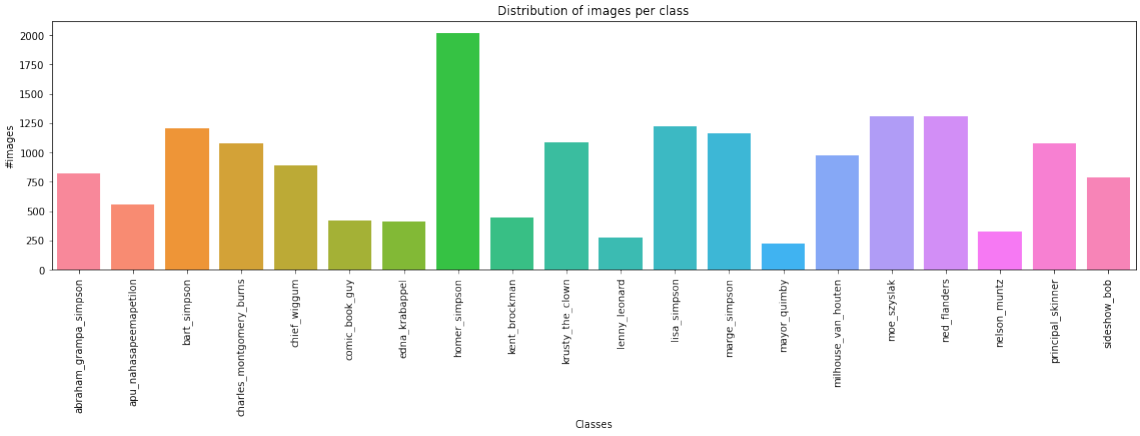

Fig. 1. Barchart showing distribution of number of images in each category

One major issue faced with the dataset was the imbalance of the categories. The number of samples per category ranges from 2022 for Homer Simpson, to 222 for Mayor Quimby. Both under sampling and oversampling of categories can lead to mis-categorisation and negatively affect the model's performance. The full breakdown of graphics is shown in figure 1. https://www.overleaf.com/project/60bad3141babf70a5570146d

In-order to overcome this issue we have could choose either to increase the number of samples in under sampled regions or decrease the number of samples in oversample regions. We opted for the former and increased the amount of data we had by generating more samples in the most underrepresented categories. To do this we identified all categories whose image count was less than a specified threshold. We then used our custom data augmentation function to generate new samples, until the categories all reached the threshold. For this threshold we chose 0.5 standard deviations below the mean.

This resulted in a total of 23829 images including the augmented images that we would use to train our final model.

2.2 Network Architecture

Initially we experimented with building our own 5-layer convolution network in TensorFlow [1] using RELU activation, and incorporating batch normalization, max pooling and dropout to mitigate overfitting. Our best performing model for this resulted in an 85% validation accuracy, but only a 71% testing accuracy, which shows clear evidence of overfitting.

To improve model performance, we implemented 2 pre-trained models available in the Keras library, InceptionV3 [6] and Xception[2], as both have relatively few trainable parameters while still having high accuracy on the Imagenet Classification challenge[5]. As time and resources were limited finding models with fewer trainable parameters was necessary as it would allow us to train multiple variations while fine tuning hyper-parameters. Additionally, fewer trainable parameters is advantageous as it results in a more robust model, that is less prone to overfitting.

Both models were used out of the box, as provided by the Keras library. No changes were made to the existing architecture and we simply added global average pooling, dropout, and batch normalization before our classification layer to help reduce overfitting.

2.3 Hyper-Parameter Tuning

One of the biggest hurdles in machine learning is choosing an appropriate learning rate. Too small a learning rate and model convergence is slow; too large a learning rate and the model may never converge, or overfit rapidly. To overcome this, it is common practice to use a learning rate scheduler, which incrementally reduces the learning rate, or to use an optimizer with an adaptive learning rate, such as Adam [3]. We choose the latter approach for all experiments discussed in this paper.

As we were using a pretrained model, and an optimizer with adaptive learning rate, this limited the number of hyper parameters for us to tune significantly. The two most impactful hyperparameters we found were the dropout rate before our prediction layer, and the image size.

In optimizing image size we found that larger image size lead to improved model accuracy but decreased computational efficiency. We found 180x180 pixels to be a good balance between speed and accuracy. After some experimentation we found dropout of 0.5 to be optimum for this model.

For. our final model we settled on InceptionV3 with a dropout of 0.5 and 180x180 image size. To train the InceptionV3 model we used a 2-step process. We first froze the upper layers of the model, then trained only the added layers for 15 epochs, at which point accuracy had converged. We then unfroze the inception upper layers to finetune them to our data and ran this for 35 epochs when it converged. We were somewhat conservative with the number of epochs to avoid overfitting.

3 RESULTS

Our final development accuracy with this model was 96.2% and our final accuracy was 96.5%. We did, however, see several characters mis-categorized in our validation testing. While this is an improvement from previously, underrepresented characters are still performing poorly as shown in figure 2.

As we can see there is still evidence of mis-categorisation of samples, namely, Nelson with 83% successful identification, Enda with 93%, and Lenny also with 93%. All 3 of whom were identified in our 6 most under sampled categories.

Interestingly Homer has middling performance, despite being the most sampled character. This is due to other character being incorrectly labeled as him.

4 DISCUSSION AND CONCLUSION

As mentioned above a main source of error in our model was miscategorization of under sampled, and over sampled characters. In future it would be ideal to source additional data as well as use data segmentation, to help even up the number of samples in each category. Given the high miscategorization of Homer Simpson, the most abundant category, it might have been constructive to remove some of the Homer samples. This highlights the importance of having as close to a uniformly distributed number of samples in each category as possible.

Overall we conclude that InceptionV3 pretrained on ImageNet, is a fast and efficient model for simple computer vision problems such as this. We found data augmentation to be an effective way to reduce overfitting, but ineffective as a means of bolstering under sampled categories. Utilising a high dropout rate will slow convergence of a model but improves the robustness, allowing it to perform better on unseen data, as will higher image resolution. As such we would recommend using the highest resolution images you can find.

The source code for our work can be found at https://github.com/SamAnast/Deakin-AI-Challenge

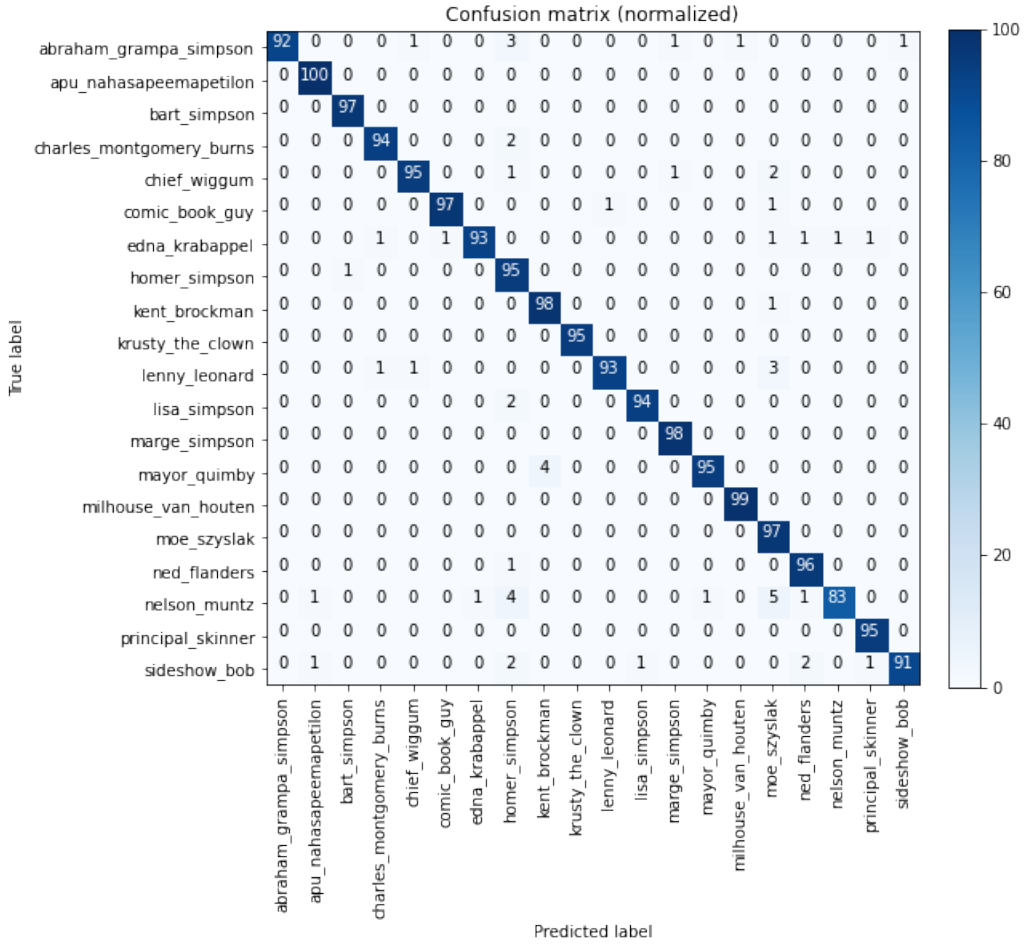

Fig. 2. Confusion Matrix showing validation results of InceptionV3 Model trained for 35 epochs.

REFERENCES

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)* (Savannah, GA, Nov. 2016), USENIX Association, pp. 265–283.

[2] Chollet, F. Xception: Deep learning with depthwise separable convolutions, 2017.

[3] Kingma, D. P., and Ba, J. Adam: A method for stochastic optimization, 2014.

[4] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems 25* (2012), 1097–1105.

[5] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *International journal of computer vision 115*, 3 (2015), 211–252.

[6] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision, 2015.