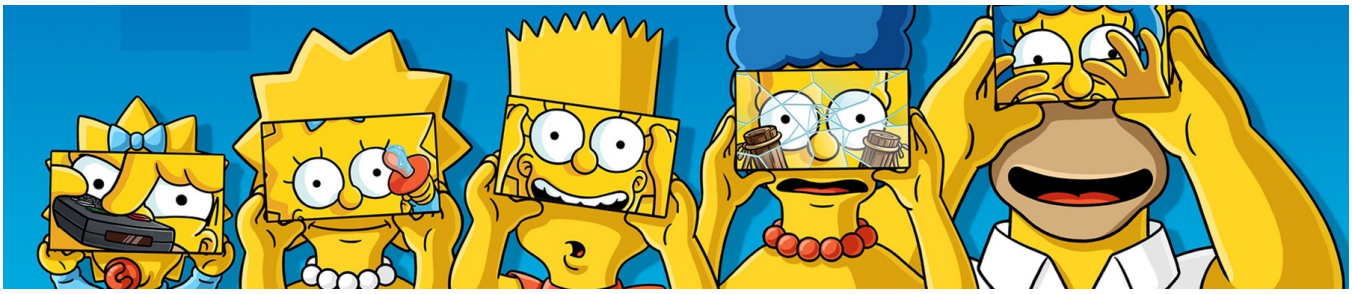
About Face: Computer Vision for The Simpsons

Josh Anthony
Deakin University
Melbourne, Australia
josh@joshanthony.net

Roger Middenway
Deakin University
Melbourne, Australia
rmiddenway@gmail.com

Melissa Mony
Deakin University
Melbourne, Australia
melissamony@gmail.com



Simpsons Faces by K. Tokis

ABSTRACT

Facial recognition technology is becoming increasingly pervasive in our modern digital lives, from video surveillance to social media, our faces are constantly tracked, classified, and stored for recognition purposes. Artificial intelligence powers these processes using carefully constructed models trained on datasets to recognise people and objects. Many data science techniques can be highly theoretical when taught at university, prompting the creation of the Deakin Simpsons Challenge to provide an opportunity for students to apply these techniques in practice to solve a real-world problem. In this report we present our approach to this challenge using transfer learning, hyperparameter optimisation and data augmentation. Our team's winning submission achieved 4th place in a competition that received over 400 submissions from a variety of undergraduate and graduate students in the school of IT. Code for our approach can be found here: https://github.com/joshanthony/simpsons-machine-learning-challenge

KEYWORDS

computer vision, facial recognition, transfer learning, convolutional neural networks

1 INTRODUCTION

Computer vision is an exciting area of artificial intelligence that aims to replicate the human visual system. The focus of this report is on facial recognition and machine learning which can be used to train a high-level understanding into a model so that it can automatically classify and recognise faces within an image. This technology is seeing increasing real-world use-cases in surveillance, smartphones, and social media [3].

The Deakin Simpsons Challenge 2021 is a machine learning competition with the aim to develop the most accurate model to recognise individual characters from The Simpsons tv show [1]. The provided dataset consists of 19548 labelled images of 19 different characters that participants can use to train their model. Submissions are then uploaded on CodaLab for evaluation, measuring performance using an accuracy classification score on a private test set. The challenges posed by this problem revolve around hardware limitations for training the model and over-fitting to the training dataset, resulting in a model that is too specialised.

This report will outline our strategy for training an effective model using transfer learning, hyperparameter optimisation and data augmentation. Using Tensorflow, the convolutional neural network will form the basis of our facial recognition system by extracting features within the images to generate feature maps to be fed into the dense layers of the neural network and match them to labels [4]. Our strategy to implement transfer learning will allow us to capitalise on models that have already been trained on massive datasets by incorporating these pre-trained layers within the model architecture alongside more specialised layers targeting the problem.

2 APPROACH

Our approach was to use an original set of neural network layers connected to an existing model that has been pre-trained on the popular Imagenet database in a process called transfer learning and fine tuning. Imagenet includes over 14 million images, many with similar features to the images that exist within our dataset, which meant that Transfer Learning provided greater efficiency so the team could allocate more time for testing and fine tuning of hyperparameters.

The pre-trained base model we selected was the EfficientNetB4 model, which is a scaled version of EfficientNet created by Google AI [6]. We unfroze a number of EfficientNetB4 layers and added several original extensions to EfficientNetB4 in order to achieve a balance between accuracy and training efficiency.

2.1 Model selection and extension

Before we selected EfficientNetB4 we spent several weeks testing and fine tuning other pre-trained models including popular variations of VGG and ResNet [5] [2]. EfficientNetB4 with the last block of 31 layers unlocked was our final model choice because it had the

best overall accuracy compared to other pre-trained existing models. During this model selection process we designed and started testing an original set of neural network layers as an extension to the base pre-trained model. Our initial high level architecture is shown below.

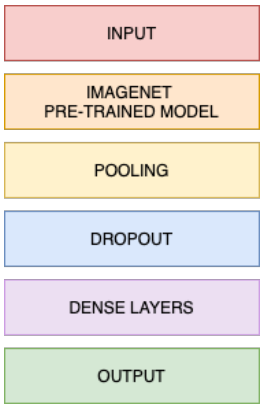


Figure 1: High level architecture of our model

The use of transfer learning allowed us to achieve greater accuracy and efficiency because the existing model recognised features that were similar to our Simpsons Dataset allowing our team to allocate more time for testing and fine tuning of hyperparameters.

2.2 Hyperparameter tuning

After we had developed an initial architecture for our model we conducted additional experiments using the Keras Tuner library to perform random search over hyperparameters and extensions to the pre-trained base model [7]. This approach was time and GPU intensive but allowed us to achieve a range of optimisations to the extension of our base model in areas such as global pooling (average or maximum), amount of dropout (0 - 0.6), number of dense layers (1-4), activation functions (ReLU, Softmax and Sigmoid) and optimisation function (Adam, RMSprop and SGD).

At the completion of this process we settled on a architecture that was able to best serve the requirements of this challenge as shown below. We also closely monitored and adjusted learning rates and took steps to reduce the learning rate in the event of a training plateau.

2.2.1 *Final custom model composition.*

- Input size 224
- EfficientNetB4 with final block of 31 layers trainable
- Global average pooling
- 0.05 dropout
- Dense ReLU layer
- Dense softmax layer
- Output with Adam optimizer

2.3 Fine tuning

Once our model was performing in the range of 98.92% accuracy we began the process of fine tuning the model by unlocking all layers excluding batch normalisation layers and trained the model on a low learning rate of 0.0001 or lower. This allowed us to achieve a final model accuracy of 99.24% which was validated by CodaLab at ~99%.

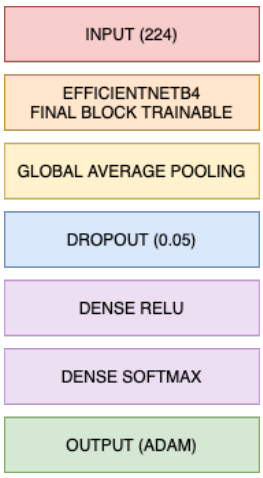


Figure 2: Specific final architecture of our model

2.4 Data cleaning and augmentation

One final aspect of our model's performance was the quality and variability of the data used for training and testing. During preliminary analysis we discovered that several classes were not well represented because they had mislabelled or duplicate data, or a low number of samples. Overall, 34 samples were found to be clearly mislabelled, and a further 26 samples contained multiple characters.

Several strategies were used to clean and augment our data. In order to remove duplicates, a script was created which generated a hash of each image using the imagehash library, and then compared each hash against the others of its respective class. For each image pair with a hash difference of less than 2, one image was removed. Mislabelled images detected by visual inspection were also removed. Some manual data augmentation was carried out, using google image search and screen shots of streamed Simpsons episodes. This augmentation focused on increasing the pool of images for comic_book_guy, lenny_leonard, mayor_quimby and nelson_muntz, all of which were both underrepresented, and on the low end of the f1 score spectrum. The addition of this data helped our model better generalise for those specific characters.

Finally, we amplified the base rotation, angle, flip and shift present within the ImageDataGenerator class used by Keras for random data augmentation during training and this helped our model generalise better.

2.5 Hardware

The model was trained using a NVIDIA Quadro P5000 GPU with 16 GB GDDR5X memory virtual machine from Paperspace. Additional experimental models were trained using GPU's in Google's Colaboratory development environment.

3 RESULTS AND DISCUSSION

Our final model obtained F1 scores ranging between 98% and 100% across all the classes. It was initially noted that the underrepresented classes tended to perform worse than the more dominant ones. The first plot in Figure 3 clearly shows this effect, with support levels under 75 being strongly associated with low scores in precision, recall and F1.

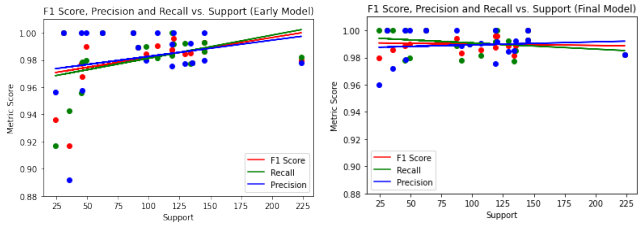


Figure 3: Performance metrics by support before (left) and after (right) data augmentation

In augmenting our dataset, we focused on collecting more samples for the four most underrepresented classes. After further training on the augmented dataset, our model showed a marked improvement for these classes. As demonstrated in the second plot in Figure 3, the classes with low support now have improved scores in all metrics, and the correlation between these metrics and support is now insignificant. (Note: these tests were performed on the raw dataset, so the supports indicated in Figure 3 are not indicative of the class balance in the augmented dataset).

Our data cleaning strategy was less successful, showing insignificant improvement despite the removal of several hundred images, including duplicates, near-duplicates, and mislabelled images. This may be explained by the capacity of neural networks to handle imperfect training data. The presence of duplicates in the dataset was also likely not a problem due to the random transformations applied to the dataset during training.

Figure 4: Mislabelled images from the dataset

The difficulty in improving the model beyond an accuracy score of 99% may be explained by several factors. Firstly, errors and ambiguities in the labelling of the dataset set a hard ceiling on the achievable accuracy. In the manual review of the dataset, 34 mislabelled images were found, as well as 26 images which contained several characters (Figure 5) and which effectively reduced the model's accuracy to a coin flip. These 60 images represent approximately 0.3% of the full dataset, thus setting a global achievable accuracy cap of around 99.7%. Another confounding factor was the presence of images in which characters were partially obscured or depicted in a highly atypical outfit or position, and which were not easily identifiable by visual information alone.

Figure 5: Images with multiple characters visible

Figure 6: Images with distorted or highly atypical representations of characters

As seen in the final results in Figure 7, the model's F1 scores varied between 98% and 100%, with generally no clear intuitive link between character and performance. While apu_nahasapeemapetilon's F1 score of 100% could be attributed to the fact that he's the only dark skinned character in the dataset, it is unclear why more generic looking characters such as mayor_quimby and ned_flanders achieved similar performance. Another interesting finding was the mediocre performance of some large classes such as bart_simpson and lisa_simpson. This may be because their status as main characters sees them shown in a wider variety of costumes and situations than other characters. It may also simply be because they are objectively less recognisable, and that a human observer finds them more readily recognisable due to a priming effect.

	precision	recall	f1-score	support
abraham_grampa_simpson	1.00	0.99	0.99	84
apu_nahasapeemapetilon	1.00	1.00	1.00	57
bart_simpson	0.99	0.97	0.98	127
charles_montgomery_burns	0.98	1.00	0.99	112
chief_wiggum	0.99	0.99	0.99	88
comic_book_guy	1.00	1.00	1.00	46
edna_krabappel	1.00	0.98	0.99	41
homer_simpson	0.99	0.98	0.99	215
kent_brockman	1.00	0.98	0.99	43
krusty_the_clown	0.99	1.00	1.00	111
lenny_leonard	1.00	1.00	1.00	31
lisa_simpson	0.98	0.98	0.98	128
marge_simpson	0.98	0.98	0.98	122
mayor_quimby	1.00	1.00	1.00	32
milhouse_van_houten	1.00	0.99	0.99	96
moe_szyslak	0.98	1.00	0.99	128
ned_flanders	0.99	1.00	1.00	134
nelson_muntz	0.98	1.00	0.99	42
principal_skinner	0.99	0.98	0.99	107
sideshow_bob	0.99	1.00	0.99	80
accuracy			0.99	1824
macro avg	0.99	0.99	0.99	1824
weighted avg	0.99	0.99	0.99	1824

Figure 7: Classification report of final model

4 CONCLUSION

In this report we outlined our approach for the 2021 Deakin Simpsons Challenge. We opted to "stand on the shoulders of giants" through our implementation of transfer learning. This allowed us to expedite the learning process and shift focus to the experimentation of parameters in our own specialised layers targeted to the problem. Alongside data augmentation this approach helped us achieve 4th place on the final challenge leader board.

ACKNOWLEDGMENTS

The authors acknowledge the support of Dr. Mohamed Reda Bouadjenek, Dr. Sunil Aryal and Prof. Peter Eklund from the School of Information Technology at Deakin University.

REFERENCES

[1] Mohamed Bouadjenek. 2021. *Deakin Simpsons Challenge 2021.* Retrieved 4 June, 2021 from https://github.com/rbouadjenek/deakin-simpsons-challenge2021
[2] Shaoqing Ren Kaiming He, Xiangyu Zhang and Jian Sun. 2020. Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2020).
[3] Thorin Klosowski. 2020. *Facial Recognition Is Everywhere. Here's What We Can Do About It.* Retrieved June 4, 2021 from https://www.nytimes.com/wirecutter/blog/how-facial-recognition-works/
[4] Laurence Moroney. 2020. AI and Machine Learning for Coders. (2020).
[5] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. *ICLR* (2015).
[6] Mingxing Tan and Quoc V. Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *International Conference on Machine Learning* (2019).
[7] Keras Team. [n.d.]. *Keras Tuner.* Retrieved June 3, 2021 from https://github.com/keras-team/keras-tuner/