MSIA-432
Deep Learning

The Simpsons Recognition Project

June 4, 2020

Siqi Li | Lirong Ma | Rocky Ye | Haonan Zhang | Nancy Zhang | Luping Zhao

Table of Contents

Problem statement

The Simpsons is a popular American animated sitcom created by Matt Groening and has been on air since 1989. As big Simpson fans, we enjoy watching the across-the-board comedy masterpiece to relax and have fun. However, because of the large number of characters, sometimes we feel like we have seen a certain character before but don't know exactly who she or he is while watching the show. This project adopted advanced convolutional neural networks VGG16 / VGG19 (Simonyan & Zisserman 2014) and Xception (Chollet, 2017) to classify images for 18 Simpson characters. In addition to image classification, this project further trained a Faster R-CNN, an object detection model, to detect and classify images with multiple characters. The deployment of this project will allow viewers to know which Simpson characters they are watching without pressing pause to check their phones.

Dataset

Dataset Description

The Simpsons Characters Data is a public data set created by Alexandre Attia. It contains images of Simpsons characters directly taken from TV show episodes and labeled by the author. The raw data contains 20,933 pictures (~45kBeach) for 48 characters. As some characters don't have many pictures, we purposely selected top 18 characters from 48 available characters in the original data set to make sure we have enough training data (more than 350 pictures) for each character. This gave us a total of 18,992 samples. For these 18 characters, we divided pictures into 3 datasets: Training set (60%), Validation set (20%) and Test set (20%).

Examples at various levels

Easy

There are not many distractions in the background. Character's face is completely shown in the middle of the picture.

Medium

Only part of the character's face is shown in the picture.

Hard

We have some pictures with characters wearing a mask, pictures with other elements in the background, and pictures with characters showing special facial expressions. These pictures may confuse the model and make it hard for the model to make a classification.

Challenges

Wrongly labeled pictures

Few images are labeled incorrectly – for example, the following character is Lisa Simpson, but it is labeled as Marge Simpson in the data set. Because of the size of the data, it's hard for us to check whether every single image is labeled correctly.

Missing labels images with multiple characters

Some images include multiple characteristics, which are great use cases of doing multiple objects detection. Unfortunately, the Simpson dataset only contains one character-label for each image. Under this circumstance, the team considers evaluating the characters with the largest probability of each plot and see whether the object detection algorithm can detect the most prominent characters of this image.

Technical Approach

Minimum Viable Product (MVP)

- A small subset of data: 18 characters, each with 200 pictures
- Resized images to (32,32) without any image augmentation.
- Model: VGG16
- Accuracy: 64.35% in the test set.

Data Augmentation

To improve model performance, we augmented the training data by

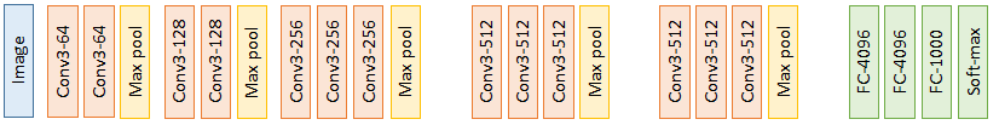
- Randomly rotate images (rotationrange = 15)
- Randomly zoom inside pictures (zoom_range = 0.2)
- Randomly apply shearing transformations (shear_range = 0.2)
- Randomly flip images horizontally/vertically
- Randomly shift images horizontally/vertically (shift_range = 0.2)

Part 1: Image Classification with Convolutional Neural Network (CNN)

Image classification via Convolutional neural network plays a major role in image processing, since it uses multiple filters to learn intricated feature embeddings from the original pixel under convolutional layers and adopts pooling to summary a patch of image data to understand local features. Generally speaking, a deeper network can learn substantial discriminative features to classify the images when it has an appropriate structure of convolutional and pooling layers. The team adopted several CNN structures who are the top performers in the ImageNet competition each year, including vgg16, vgg19 and Xception. The detailed descriptions for these models are shown below.

VGG-16

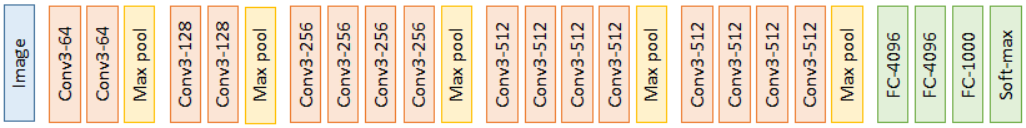
Five convolutional blocks with LeakyReLU activation functions are responsible for feature extraction. Pooling layer takes MaxPooling2D with a pool size of 2x2. The concatenation layer then converts the result from to a vector through the Flatten layer. Dropout layer subsequently excludes a randomly selected percentage of neurons and processes the output from the Flatten layer. Finally, the output layer has neurons and a soft-max activation function to output probability-like predictions for each character.



VGG-16

VGG-19

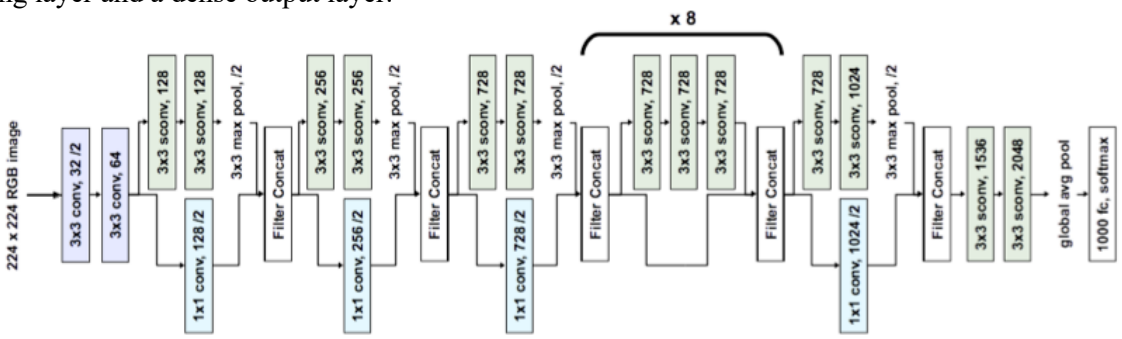
Similar to VGG-16, except it has four convolutional layers in the fourth and fifth block.



VGG-19

Xception

Xception stands for Extreme Inception, which merges the idea of GooLeNet and ResNet but replaces the inception modules with a depth-wise separable convolution layer. A regular convolutional layer uses filters that simultaneously capture spatial and cross-channel patterns. In comparison, a separable convolutional layer assumes that spatial and cross-channel patterns can be modeled separately. Hence, Xception is mainly composed of two parts: the first part applies a single spatial filter for each input feature map, and the second part, a regular convolutional layer with 1x1 filters, only focuses on cross-channel patterns. Since separable convolutional layers only have one spatial filter per input channel, it does not work well for layers with too few channels, such as the input layer. Therefore, Xception architecture starts with two regular convolutional layers and then the rest of the architecture uses only separable convolutions plus a few max pooling layers and the usual final layers, such as a global average pooling layer and a dense output layer.



Part 2: Characters detection via Faster R-CNN

In addition to image classification, the team went a step further to implement object detection models such as the Faster R-CNN. The best performer is Faster R-CNN. It mainly consists of two parts: a region proposal network and a fast R-CNN. Region proposal network (RPN) is fully convolutional network that simultaneously predicts object bounds and objectness scores at each position. The high-quality region proposals, identified by RPN, are used by Fast R-CNN for detection. In the next step, RPN and Fast R-CNN are concatenated into a single network via shared the convolutional features, and finally generate the classification for each selected region.

In this model setting, we have the following contributions:

1. Tuning hyperparameters related to initial part of feature representation under convolutional neural network, such as data augmentation, activation function, etc.
2. Considering non-maximum suppression method to avoid the RPN detector overdrawing boxes under the same regions for one character. This implementation relies on setting a threshold overlapping upper bound and lower bound, which are 0.7 and 0.3 respectively.
3. Producing the output images with the predicted boxes and reporting classification metric performance mentioned in the result section.

Results

Part I: Model result summary for image classification

The minimum viable product (MVP) that we trained at the midterm check point had an accuracy of 64.35%. To find the best model, we trained three selected models with same training set and tested them on same test set, each with several attempts of parameter tuning. Then we compared model's accuracy across all. Xception model outperformed VGG16 and VGG19 model. Its accuracy reached 97%, which is approximately a 50% increase from the MVP.

Model Comparisons

MODEL	HIGHEST ACCURACY	ACCURACY AT 100 EPOCHS	IMAGE SIZE	IMAGE AUGMENTATION	OPTIMIZER
MVP	64.35%	63.28%	(32,32)	No	Adam
VGG-16	91.08%	90.45%	(32,32)	No	Adam
VGG-16	95.61%	93.32%	(128,128)	Yes	Adam
VGG-19	90.55%	90.08%	(32,32)	Yes	Adam
VGG-19	93.11%	92.97%	(128,128)	Yes	Adam
XCEPTION	95.37%	93.74%	(64,64)	Yes	Adam
XCEPTION	96.47%	96.39%	(128,128)	No	Adam
XCEPTION	97.03%	96.66%	(128,128)	Yes	Adam
XCEPTION	96.43%	96.34%	(128,128)	Yes	SGD
XCEPTION	96.91%	96.53%	(256,256)	Yes	SGD

Best Model Performance

Model Configurations

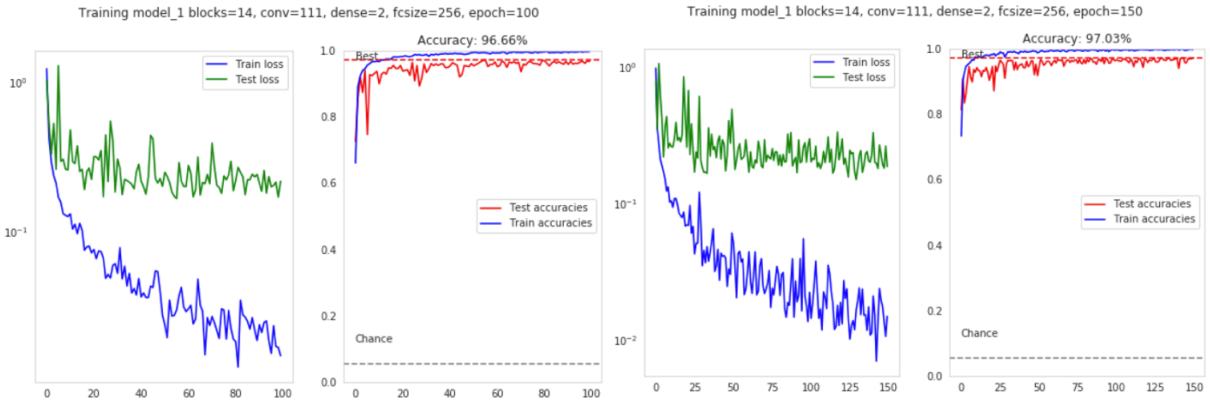
After trying VGG16, VGG19, and Xception with different parameters, we found the "best" model, which has configurations as follows:

- Optimizer: Adam

- Augmentation:zoom_range=0.2,rotation_range=15,width_shift_range=0.2,height_shift_range=0.2, horizontal_flip=True, vertical_flip=False

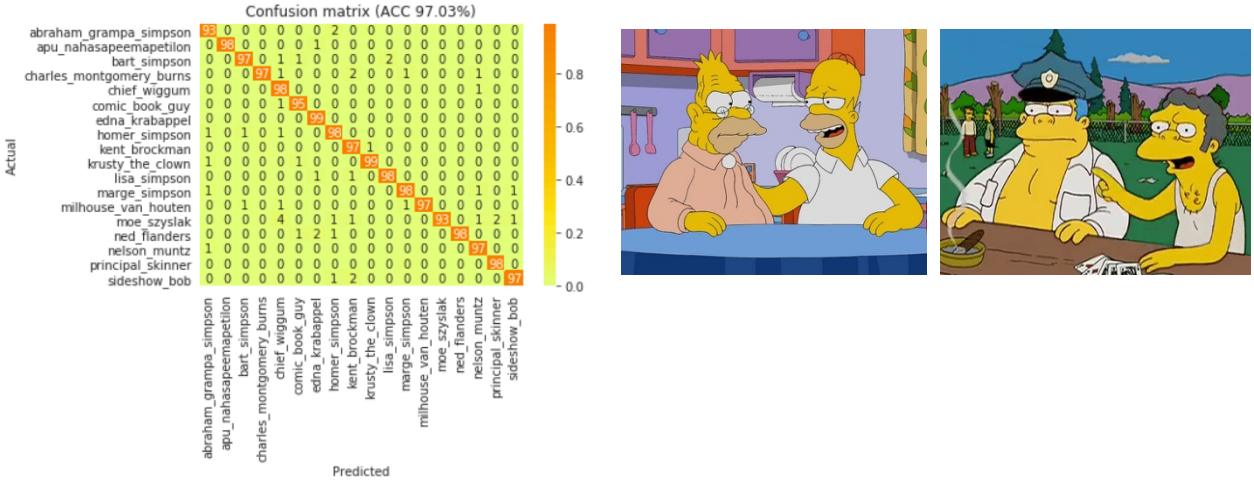
Loss and Accuracy Curves

The training loss roughly converged and became stable after 50 epochs, and the test accuracy was relatively high and stable after 50 epochs as well. The test accuracy was 96.66% at 100 epochs and achieved 97.03% at 150 epochs. Our model outperformed the models in previous work we found where the best model accuracy was around 90%.



Confusion Matrix

The confusion matrix indicates that the model has an overall good performance. Among all the characters, the model did relatively poorly to classify Grampa Simpson and Moe Szyslak. The model often confused Homer Simpson with Grampa Simpson, who are relatively similar among all the characters. As shown in the middle picture below, the colors of their mouse are the same and the shapes are similar, so the model had a hard time to distinguish them. The model also struggled to distinguish between Moe Szyslak and Chief Wiggum. As shown in the rightmost picture, certain characteristics such as their nose shapes and mouse shapes are relatively similar, which contributed to misclassification.



Heatmaps

To tell whether our model was reliable, we also looked at heatmaps. Take heatmaps for Milhouse Van Houten as an example. The two heatmaps on the left indicate that the model focused on the head, nose, and glass when classifying him correctly. The rightmost heatmap shows the model misclassified Milhouse when it was dissecting the background. Most of the time, the model was focusing on the correct part of the image, namely focusing on character faces instead of background.

Part II: Result summary for character detection

Evaluation Metrics

The evaluation metrics for character detection are classification accuracy, precision, recall and F1-score.

Classification Accuracy

The classification accuracy is defined as the total number of correctly classified boxes among all the labelled boxes divided by the total number of correctly classified boxes are the sum of the diagonal elements of the confusion matrix.

Precision

Precision measures the amount of relevant cases among all the predicted cases. For example, in order to measure the precision of Lisa Simpson, we want to see how many boxes are for Lisa Simpson (actual positive) and how many boxes are predicted to be Lisa Simpson (True positive). It measures the ability to retrieve the relevant cases under our algorithm.

Recall

Recall, also known as true positive rate, is defined as true positive cases divided by actual positive cases. For example, the recall for Lisa Simpson relies on two measures, one is the amount of correctly predicted boxes for Lisa Simpson (true positive), and another is the number of boxes with Lisa Simpson (actual positive). It indicates the ability to recall all the relevant cases from the actual cases.

F1-score

F1-score is defined as the harmonic mean of precision and recall. It gives a balanced view of model performance. The formula is 2 * precision * recall / (precision + recall). The larger F1-score, the better classification performance.

Limitation

We recognize the limitation of the dataset where the majority of the images contain only one characters. These images would be very useful cases to test the performance of our detection model. So, we decided to use precision and recall under the boxes level for each character to evaluate the performance.

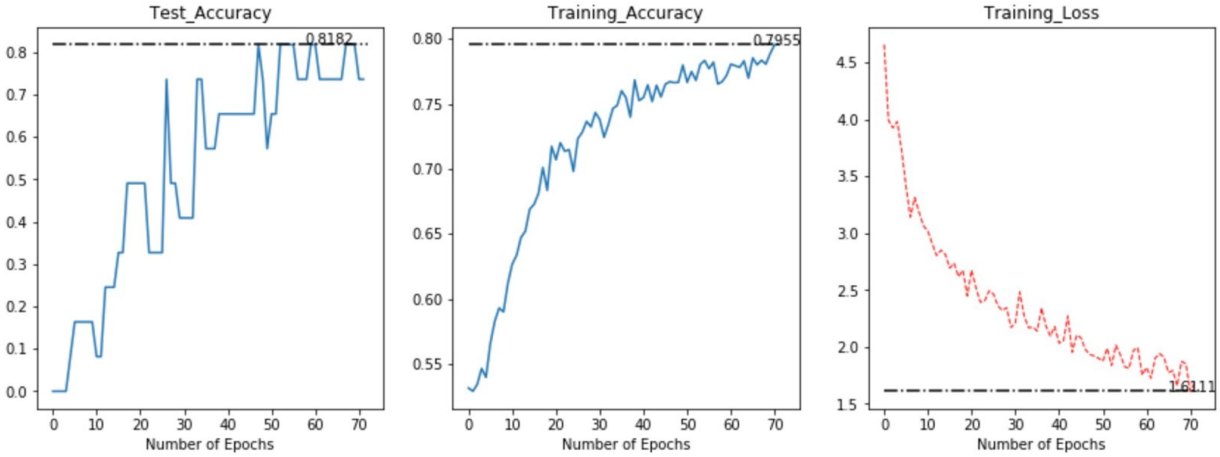
Best Model Performance

The best model is Fast R-CNN with the following model setting:

- Optimizer: Adam

- Activation function: ReLu
- Randomly rotate images: rotationrange=15
- Randomly zoom inside pictures: zoom_range=0.2
- Randomly flip images horizontally
- Randomly shift images horizontally/vertically: shift_range=0.2
- Min_overlap_region_proposal_network: 0.3
- Max_overlap_region_proposal_network: 0.7
- Number of epochs: 100
- epoch_length = 1000

The train, test accuracy curves and loss curves are shown below.



The overall performance under the validation set had an accuracy of 81.82%. When we rolled out to another unseen test set, the overall classification for boxes was around 73%, which is a reasonable classification performance under this region detection and image classification network.

The tables below are result summary tables for the Top six and Bottom Six characters under training and testing scenarios. The table includes the precision, recall and f1-score of each characteristic.

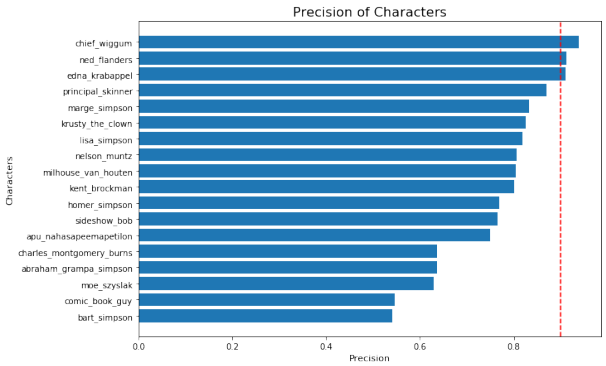
TEST METRICS PERFORMANCE - TOP SIX ACCURATE CHARACTERS

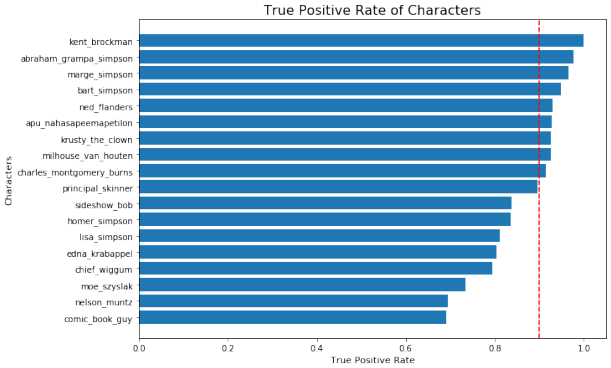
CHARACTER	Precision	Recall	F1 score
NED_FLANDERS	0.913	0.929	0.921
MARGE_SIMPSON	0.832	0.966	0.894
KENT_BROCKMAN	0.8	1	0.889
PRINCIPAL_SKINNER	0.87	0.895	0.883
KRUSTY_THE_CLOWN	0.826	0.927	0.874
CHIEF_WIGGUM	0.939	0.795	0.861

TEST METRICS PERFORMANCE - BOTTOM SIX ACCURATE CHARACTERS

CHARACTER	Precision	Recall	F1 score
ABRAHAM_GRAMPA_SIMPSON	0.636	0.977	0.771
CHARLES_MONTGOMERY_BURNS	0.637	0.914	0.751
NELSON_MUNTZ	0.806	0.694	0.746
BART_SIMPSON	0.541	0.949	0.689
MOE_SZYSLAK	0.629	0.733	0.677
COMIC_BOOK_GUY	0.547	0.69	0.611

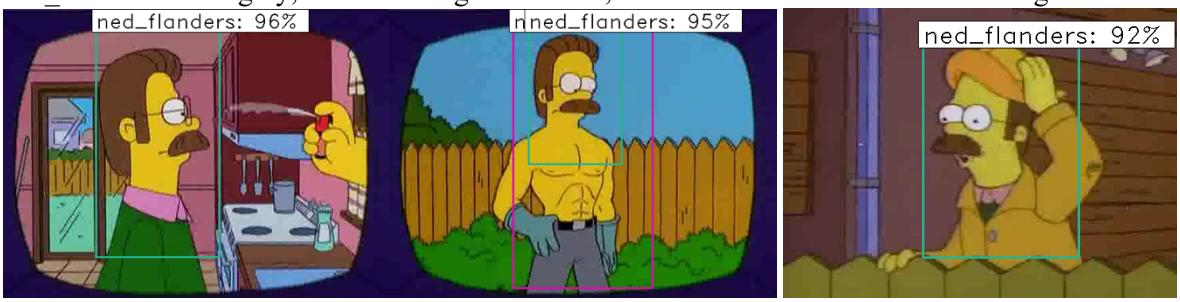
The bar plots below are the recall (True positive rate) and precision value sorted by descending orders.





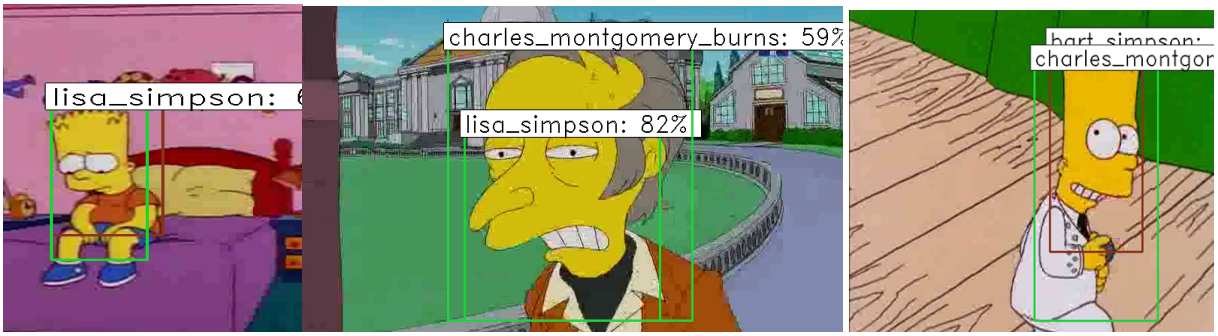
Insights

There are several insights from the metrics performance. The model sufficiently captures facial features for those characters who have salient difference from others, such as ned_flanders has the highest f1-score, with precision and recall all above 90%. The figure plots also show that the model can detect ned_flanders thoroughly, eitherfrom right-side face, front-face and left-side face with high confidence.



The same story is true for a major character marge_simpson, and the classification score for all metrics are pretty good, no matter whether marge_simpson is standing or not, with kids or not or standing at different angles. The salient features like the eyes, long blue hairs and long noise are the key regions that differentiate her from others.

However, when we look at those characters who do not perform well, for characters such as Bart Simpson, charles_montgomery_burns, abraham_grampa_simpson and moe_szyslak, they have a very high recall score, which means the model has a higher chance of finding them, but the precision score is very low. After checking the number of images in the test set, the explanation to this result involves two part: first, there are limited number of images for these characters under the test set, such as for charles_montgomery_burns, it only contains less than 30 images in this test set. So, the number of true cases is small, inflating the recall metric. Second, the model may not accurately predict the characters such as Bart Simpson and charles_montgomery_burns, as they three may sometime confound with each other based on the model training. The figures below provide supporting evidence.



The low resolution of pictures and wrongly classification largely arises from the fact that the model fails to detect these subtle differences, which can be a future area for improvement.

Conclusion and Future Work

To conclude, the classification part is highly successful. We were able to use models learned in class to produce highly accurate results. We have many of the errors due to two or more characters in the same picture. Taking that into account, we are very happy with our classification performance.

For detection we had not as good but still ok performance. We know that detection is harder than classification since you not only have to recognize characters but also have to pinpoint the exact location of the characters. Our faster R-CNN model is doing very well, and we understand that missing bounding box labels in our data contribute to a lot of the errors. Another thing we are struggling with is the long training and prediction time. On average the model takes 10min to train one epoch and one second to predict one image.

If we were to take it further, we would like to create a character detection software for people not good at recognizing characters or remembering names. We can use our algorithm to detect characters and display their names on the screen.

Future improvement. One limitation of faster R-CNN is that ironically, it's still not very fast. There is a more advanced algorithm called YOLO which standards for you only look once. it is created by Joseph Redmon. Unfortunately, he is no longer researching computer vision due to ethical concerns. YOLOv4 just came out couple of weeks ago. It is extremely fast in both training and prediction. and YOLOv4 is said to be able to achieve high accuracy. This something we would like to use in the future if we have time.

References

Attia, A. (2017, June 12). The Simpsons characters recognition and detection using Keras (Part 1). Retrieved from https://medium.com/alex-attia-blog/the-simpsons-character-recognition-using-keras-d8e1796eae36

Carremans, B (2018, August 17). Classify butterfly images with deep learning in Keras. Retrieved from https://towardsdatascience.com/classify-butterfly-images-with-deep-learning-in-keras-b3101fe0f98

Girshick, R (2015). Fast R-CNN. Retrieved from https://arxiv.org/abs/1504.08083

Ren, S et al. (2015) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Retrieved from https://arxiv.org/abs/1506.01497

He, K et al. (2015). Deep Residual Learning for Image Recognition. Retrieved from https://arxiv.org/pdf/1512.03385.pdf

Krizhevsky, A et al. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Retrieved from https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Simonyan, K and Zisserman, A (2015).Very Deep Convolutional Networks for Large-Scale Image recognition. Retrieved from https://arxiv.org/pdf/1409.1556.pdf

Szegedy, C et al. (2014). Going deeper with convolutions. Retrieved from https://arxiv.org/abs/1409.4842

Appendix - Concluding Remarks and Comments on the Project

- Do you feel that the work that you did would be applicable to a commercial environment?
  - The model for classification is ready for deployment, but it still takes longer time to train for faster R-CNN, given that the computational resource is limited.
  - The model can be applicable to a commercial environment such as the video sharing channels, which provide the audience option to choose "showing labels" when they put the cursor on a character in real-time.
- Did you feel that the extra effort spent on Deep Learning was worthwhile vs taking a simple approach? Was the extra computation beneficial to the problem, or is Deep Learning still something better suited for academics?
  - For this project, deep learning is the only solution. There is no way to classify or detect characters using models such as random forest and boosted tree. Both cloud platforms where we run our models (Google Colab and Northwestern Deepdish) support TensorFlow GPU acceleration, which speeds up the training of deep learning model. The average runtime for VGG-16, for example, is around 120 seconds per epoch.
- What do you think Deep Learning may be best suited for in your jobs/internships after this class? Do you think it'll complement R, Hadoop and Excel, or is the market not ready yet? What could make it ready if it isn't ready yet?
  - Deep learning will be suitable for the team member who are pursuing projects in computer vision (CNN) and events prediction (LSTM), especially in applied scientist and research scientist roles in technology companies. I think it will be a good complement to R and other technologies by providing more intricated feature representation, improving the approximation of true functional form, etc. But it may not be ready, since deep learning is sophisticated and requires more explanations to non-tech stakeholders.
- Which provided more value for your project: high classification accuracy or high degree of model explainability? How does this apply to other practical projects that you may be working on?
  - For this project, we value high classification accuracy more than model exploitability. The ultimate goal of this project is to correctly detect and classify character, and thus we don't care that much about how the model make the decision.
- What attributes do you think would be most helpful to seeing successful adoption of your technology? E.g. regulation, interpretability, high accuracy, speed, etc.
  - High accuracy and training speed will be the most crucial elements to the deployment of the project.
    - If classification/detection accuracy is low, the model may not be useful at all.
    - If the training of the model takes a long time due to the high number of categories or model complexity, this feature may not be available when it is needed the most (new TV shows on air).
- What can we do to help improve the curriculum?
  - More example notebooks; learning by doing; More materials on object detections
  - The course is well-structure and tailored to the students' need and interest. It would be much appreciated it if there are some real-time demos during the class. Some students may be interested in mathematical derivation, so it would be great to provide this information for supplementary documents.
- Were the computing resources adequate?
  - Set up guide for deep dish is easy to follow, but more computer power in the future could be helpful.
  - Google Colab: We tried using free version of Google Colab to run our models, but the running time for one epoch is about eight times longer compared to deepdish. Furthermore, it has a limitation on GPU time usage (12 hours for free accounts and 24 hours for pro accounts). When the model gets more complicated, it's hard to get enough number of iterations before got cut off.

  - Deepdish servers are usually overloaded and run out of memory. Adding more GPUs to existing deepdish servers will be beneficial.
- Did the assignment help you prepare for the project?
  - Yes, we use the template code from homework 2(a) to build the VGG 16 and VGG 19 models. The assignment is a good starting point for midterm evaluation, as well as final deliverable.
- Do you feel that the "AI" and machine learning aspects are helpful in MSiA? Or is visualization/presentation of the data more helpful?
  - I think both are useful in several analytics aspects. AI and machine learning are helpful because we want to get accurate prediction. Visualization and presentation of data are more useful in business analytics domain, as the effective presentation will better sell the data product and model to target audience, making them believe this black-box model will work.
- What did you learn on the project?
  - The team learns to build up advanced algorithms for image classification and detection by reading the papers and accessing source codes and interprets their result diagnostics. We cultivate great expertise on defining problem and narrowing down the focus of the project.
  - Besides, we have gained first-hand experience in optimizing the workflow, dividing tasks, and optimizing the usage of computational resources.