Deep Learning for Emotion Recognition in Cartoons

Aim

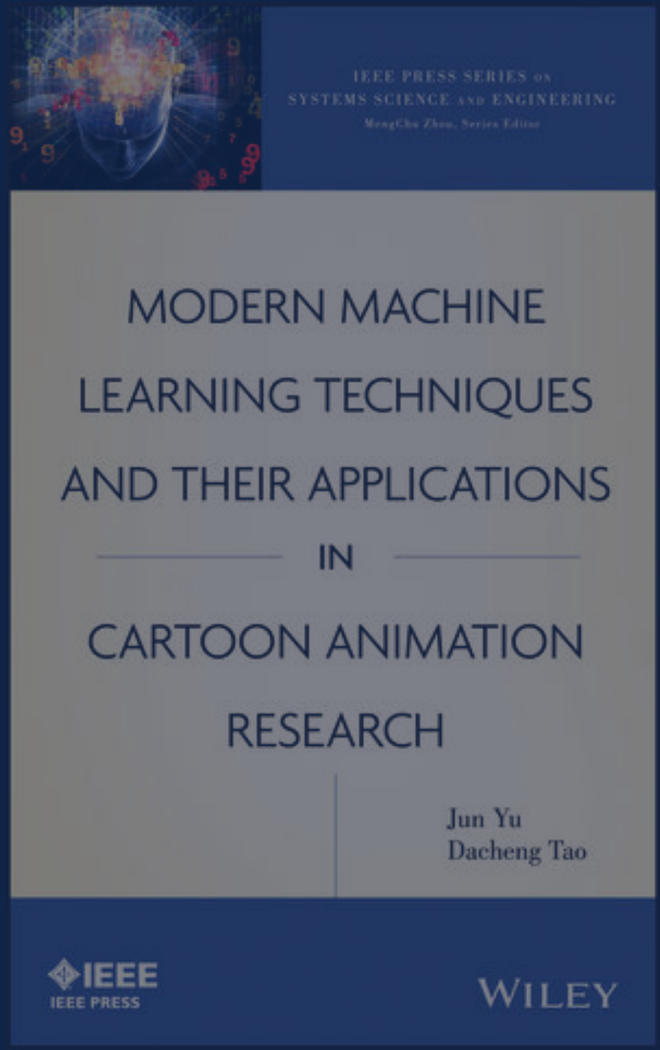
Measure how accurate the program is able to identify an emotion from a given cartoon video.

Motivation

— Current research in (facial) emotion recognition use **human faces** not cartoon faces.

— Not much research into **animated** cartoons + deep learning.

— But there *is* one book[1].



[1] Yu, J. and Tao, D. (2013) *Modern Machine Learning Techniques and Their Applications in Cartoon Animation Research*. Vol. 4. John Wiley & Sons.

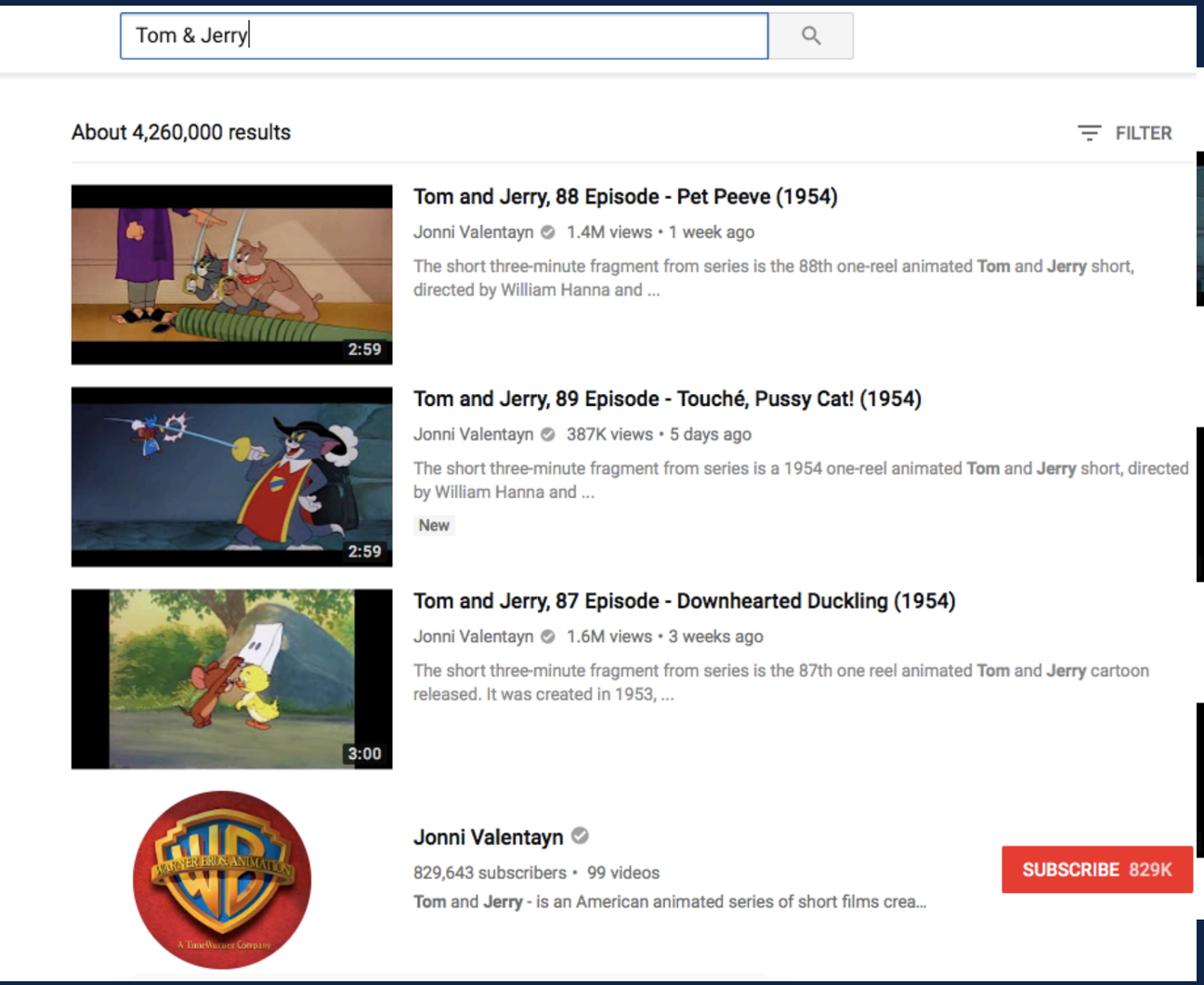
Machine Learning ≠ Deep Learning

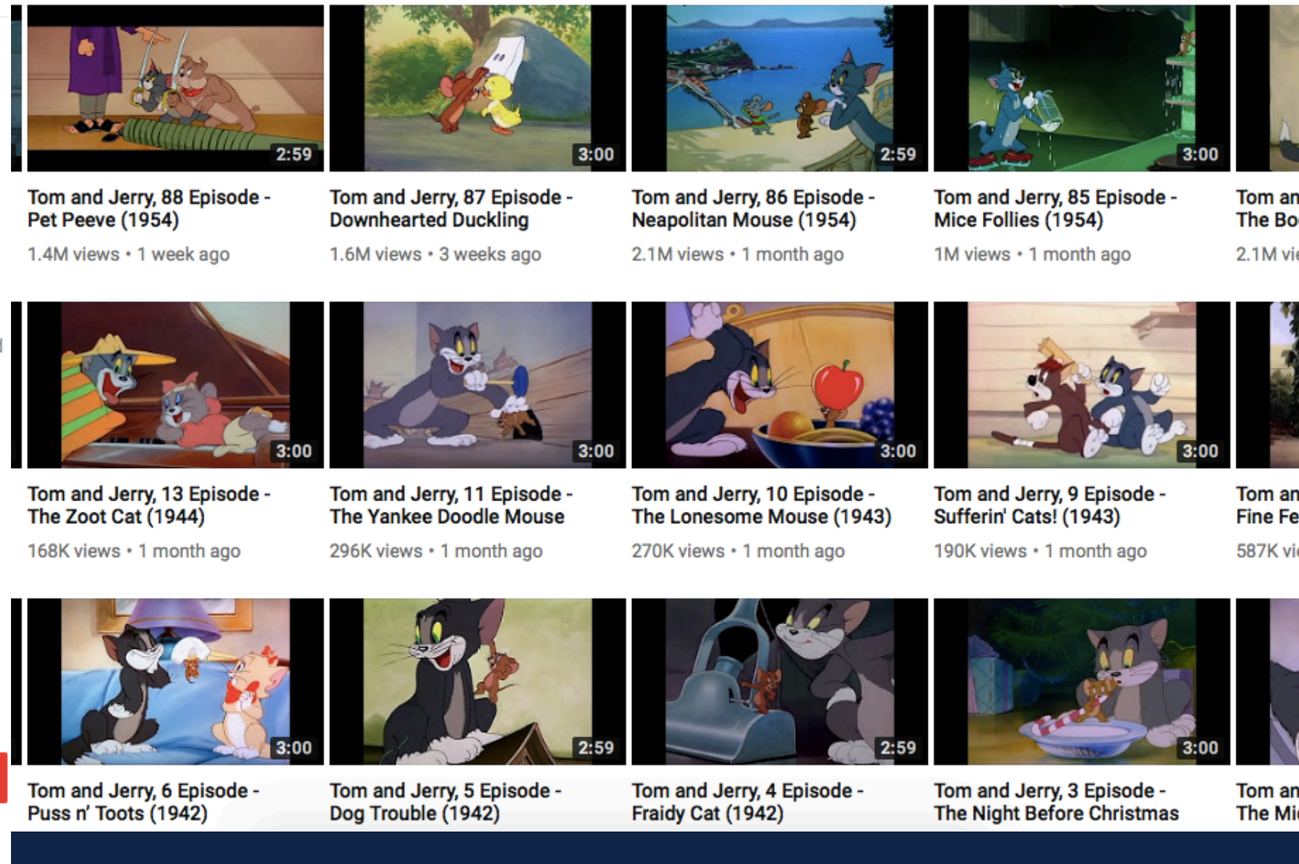
Requirements

- Choose a cartoon: Choice was *Tom & Jerry*[2].
  - Lots of various emotions in each episode.
- Segment faces from the cartoon.
- Build a dataset of emotions for each main character (**Tom** & **Jerry**)
- Train the network on a labeled dataset.

[2] *Tom & Jerry* © Warner Bros. Entertainment, Inc

Dataset Gathering

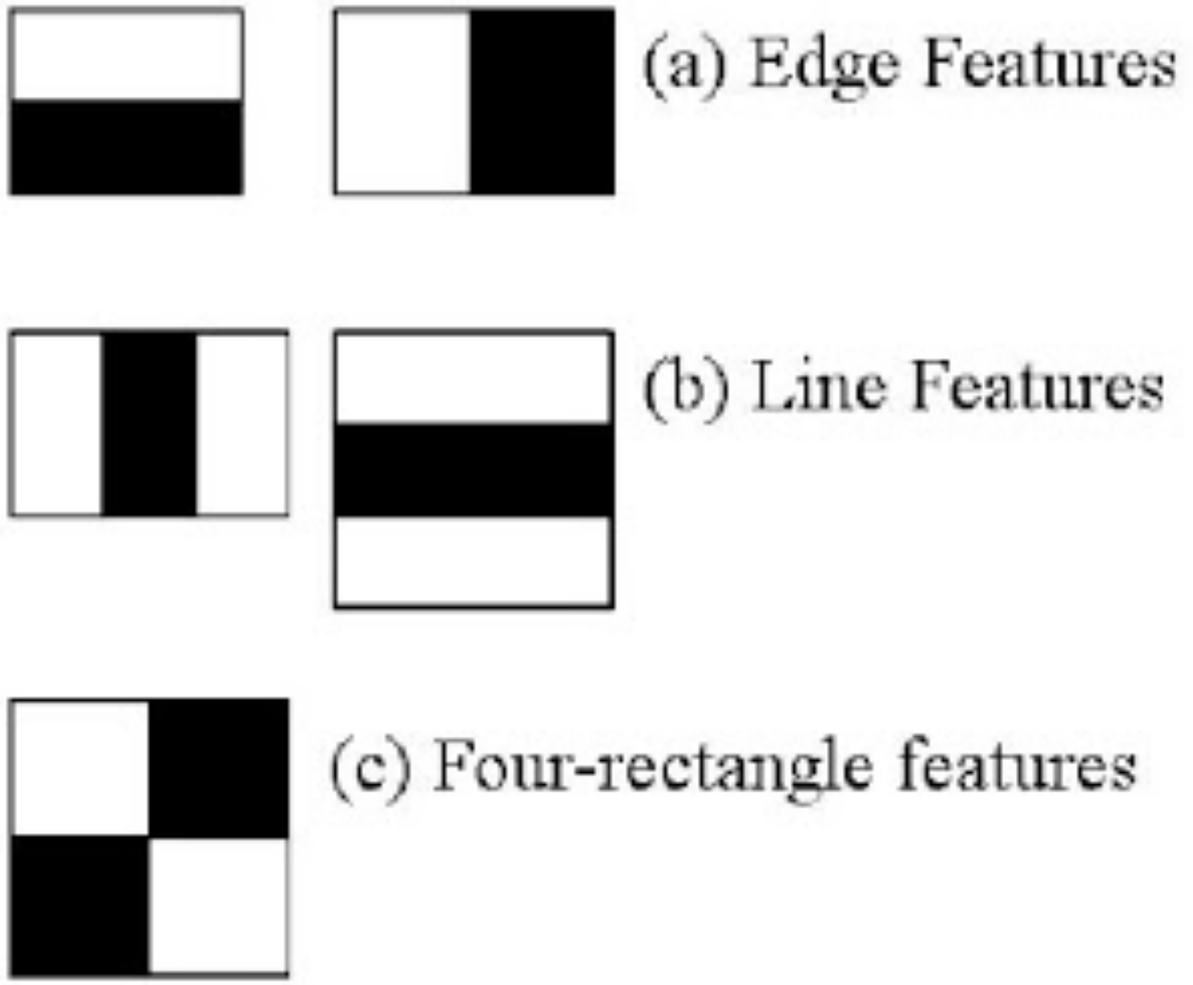


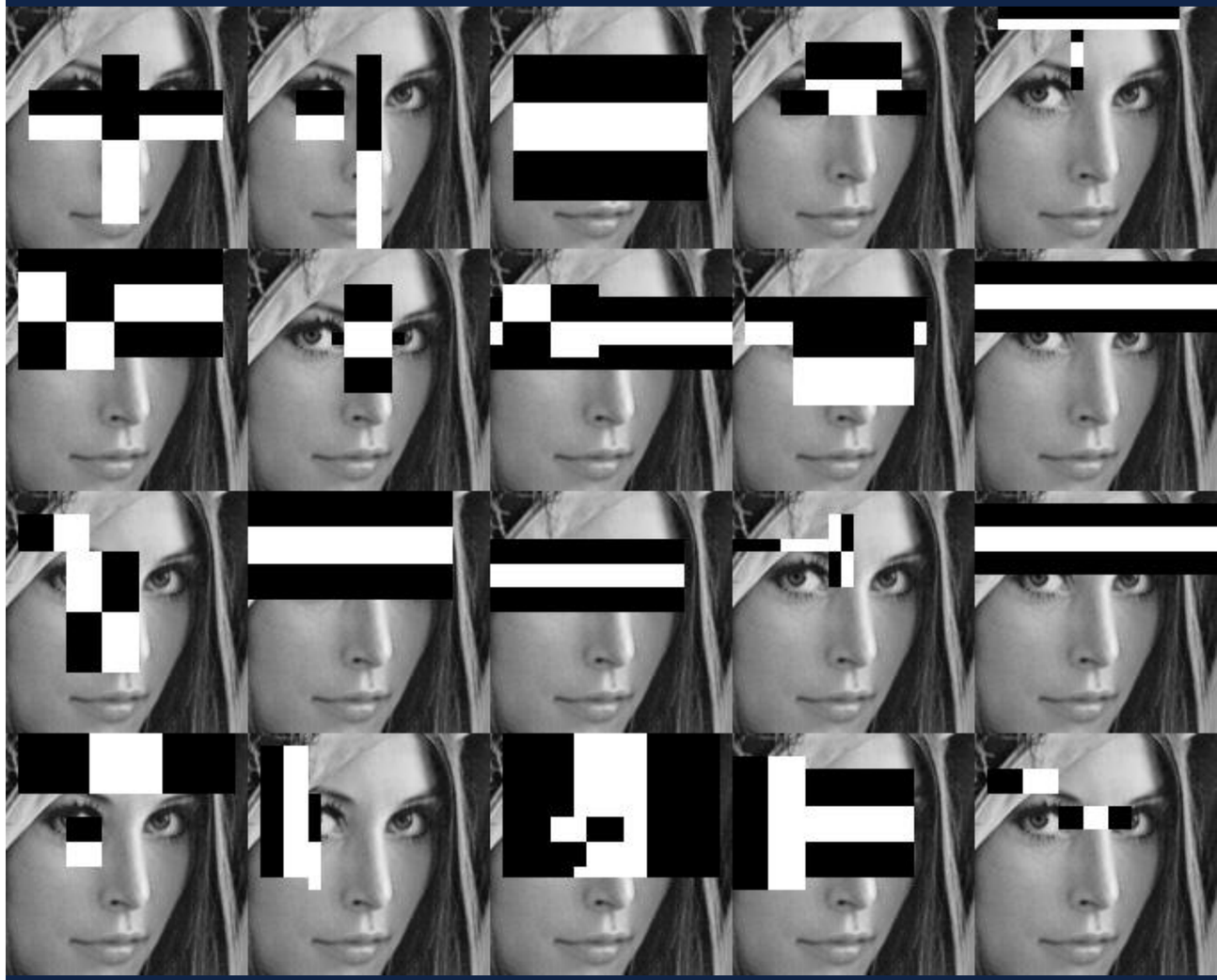


Segmentation

Haar Cascades

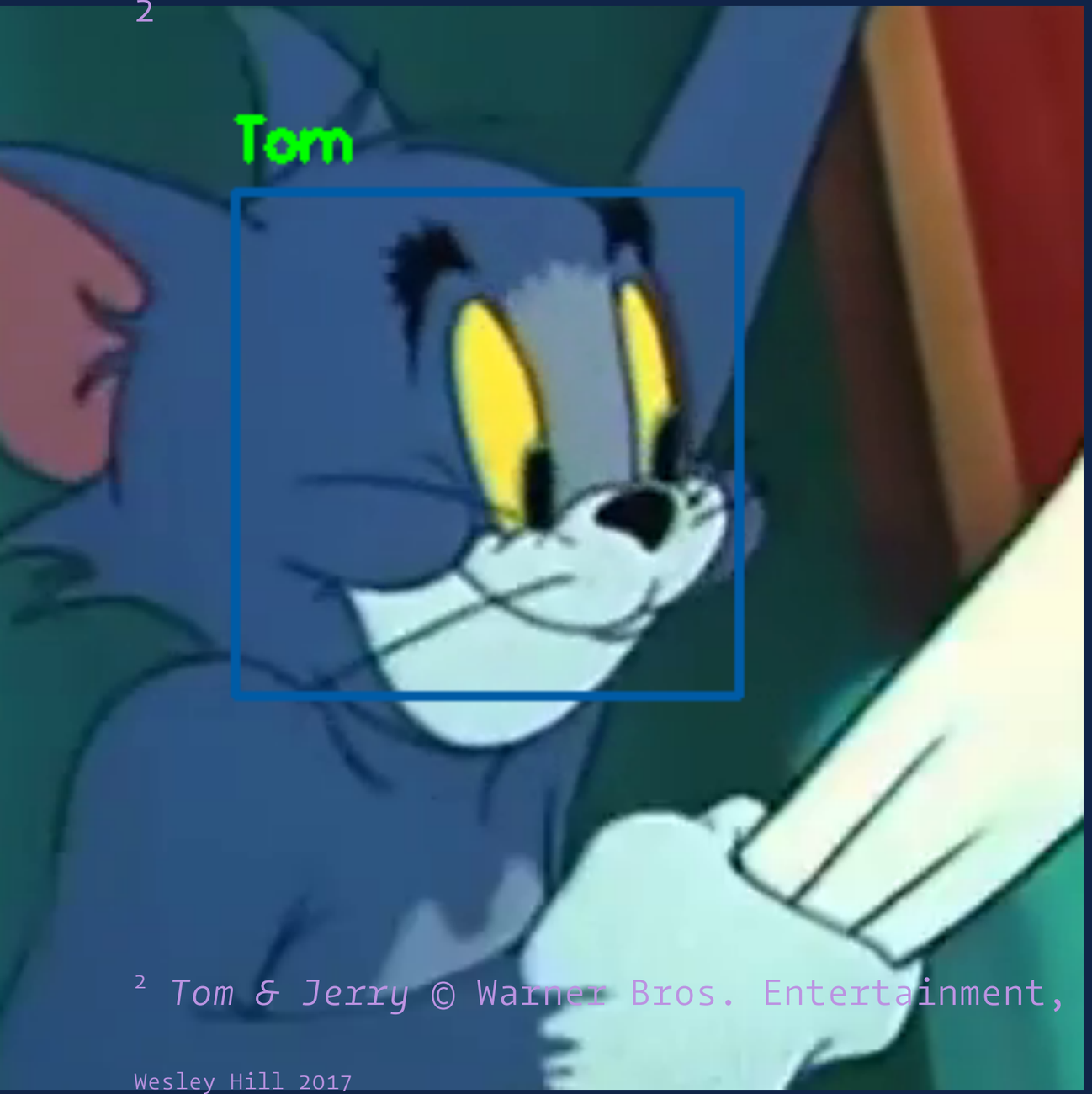
Haar Cascades

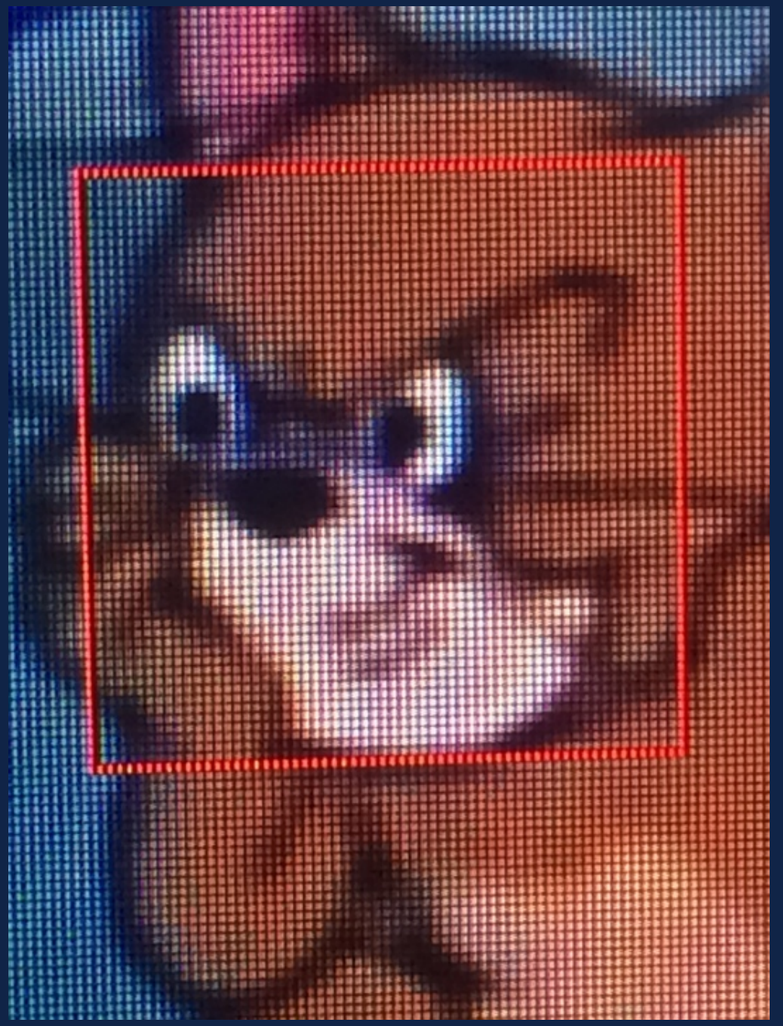




- Created custom Haar cascade for both *Tom & Jerry*.
  - There were none made online to detect cartoon faces, only human ones.
  - Depending on the window size it does detect other character faces in the cartoon.

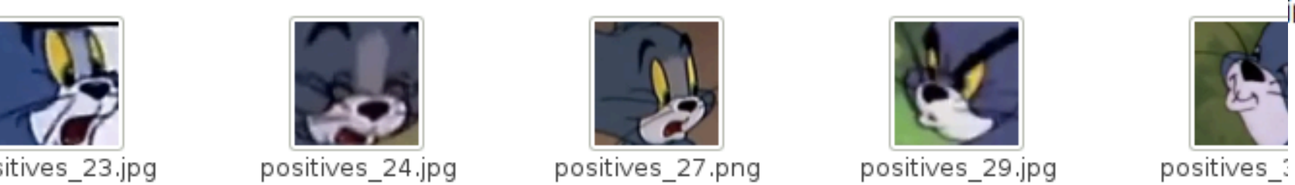
2



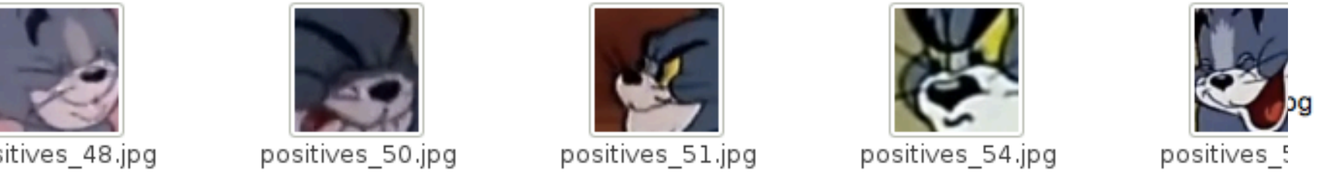


[2] *Tom & Jerry* © Warner Bros. Entertainment, Inc


positives_3.jpg positives_12.jpg positives_13.jpg positives_14.jpg positives_


itives_23.jpg positives_24.jpg positives_27.png positives_29.jpg positives_


itives_34.jpg positives_36.jpg positives_39.jpg positives_40.jpg positives_

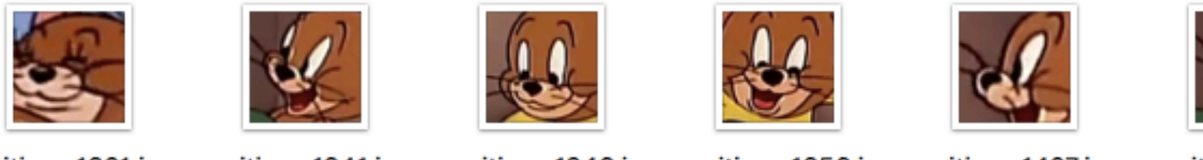

itives_48.jpg positives_50.jpg positives_51.jpg positives_54.jpg positives_

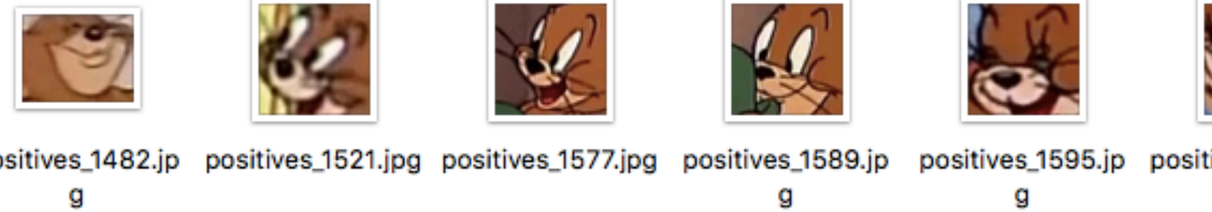

itives_64.jpg positives_65.jpg positives_66.jpg positives_68.jpg positives_


itives_76.jpg positives_79.jpg positives_80.jpg positives_81.png positives_

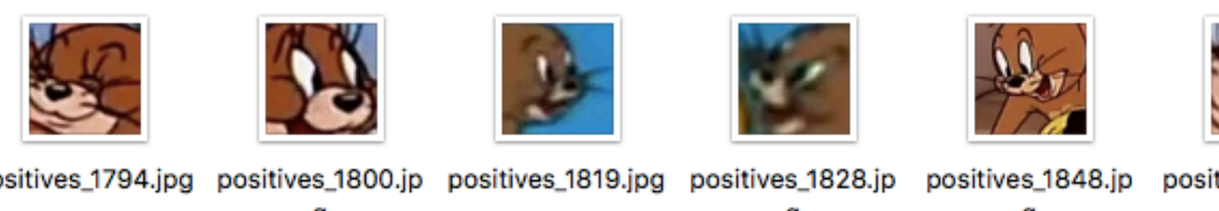

itives_88.jpg positives_89.jpg positives_90.jpg positives_91.png positives_


positives_

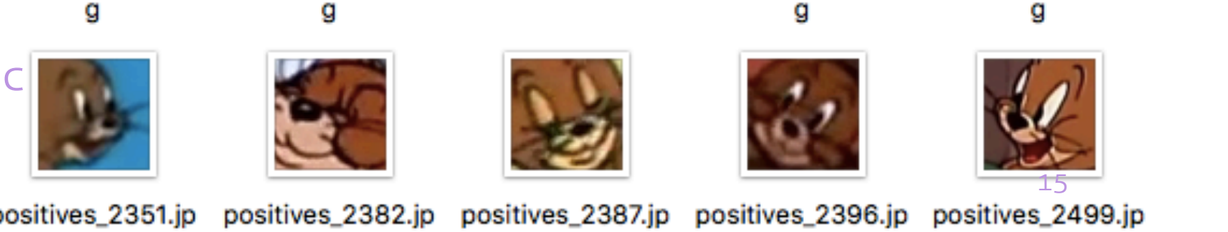

jp positives_1321.jpg positives_1341.jpg positives_1346.jpg positives_1356.jpg positives_1427.jpg posit


jp positives_1482.jpg positives_1521.jpg positives_1577.jpg positives_1589.jpg positives_1595.jpg posit


pg positives_1623.jpg positives_1642.jpg positives_1648.jpg positives_1667.jpg positives_1670.jpg posit


positives_1794.jpg positives_1800.jpg positives_1819.jpg positives_1828.jpg positives_1848.jpg posit

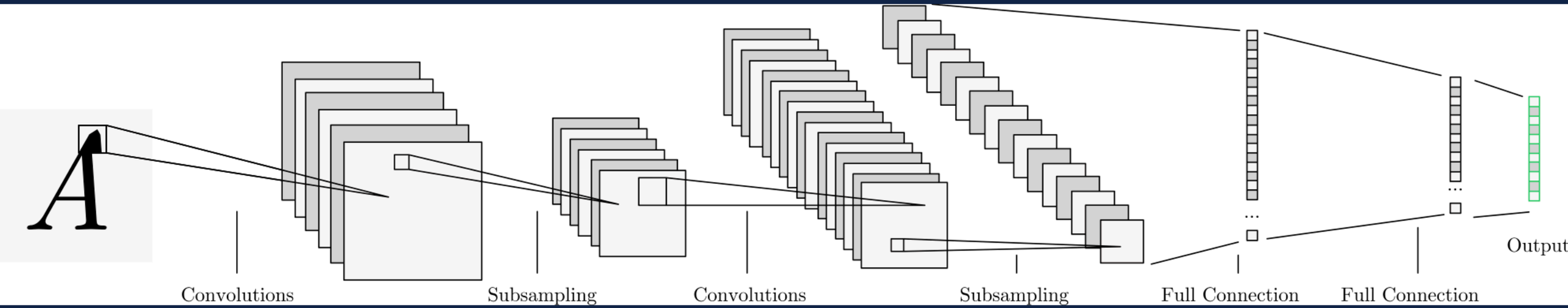

pg positives_2003.jpg positives_2096.jpg positives_2157.jpg positives_2166.jpg positives_2306.jpg posit


jp positives_2351.jpg positives_2382.jpg positives_2387.jpg positives_2396.jpg positives_2499.jpg

Dataset Stats

- In total about **159,035** images segmented.
  - For about ~64 episodes. *(Tom & Jerry has over 100)*
- Selected around 400 images for each character & emotion, (angry, happy, surprise) for training and testing.

Convolutional Nerual Network


Convolutions
Subsampling
Convolutions
Subsampling
Full Connection
Full Connection
Output

- In recent years CNN's produced great results in image & object recognition.
- The CNN is used for this project to learn **features** *(eg. smile angles, eyebrows)*.
- Framework for DL used was Keras + TensorFlow backend. (Keras also works with Theano)

- No pre-trained network.
  - Inception-V3 predicts Tom & Jerry as 'comic books'.
- Images resized to 60x60 with 3 channels. (RGB)
- 3x3 convolution & 2x2 max pooling with a input image of size **60x60x3**.

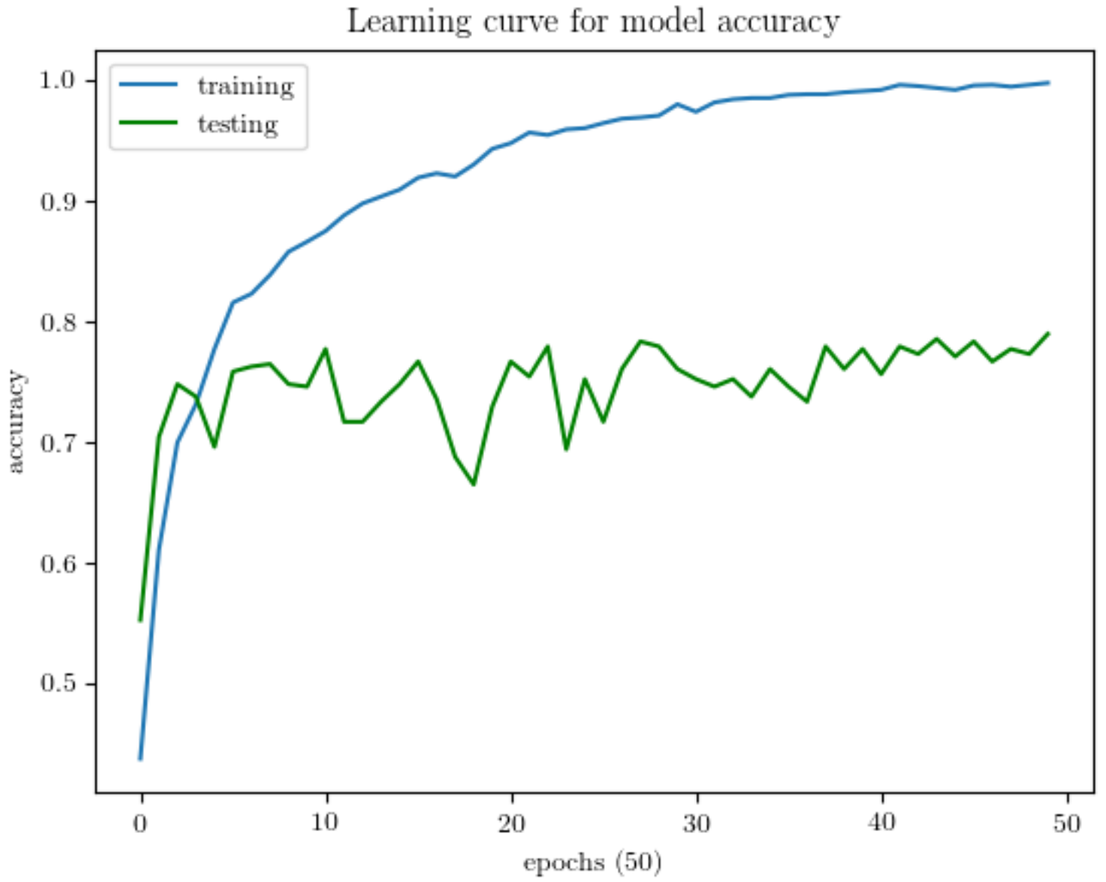
- 3x3 Convolution & 2x2 Max Pooling. (ReLU activation)
- 3x3 Convolution & 9x9 Max Pooling. (ReLU activation)
- Fully connected layer of 512 neruons.
- Final output layer of 6 neruons for each emotion.

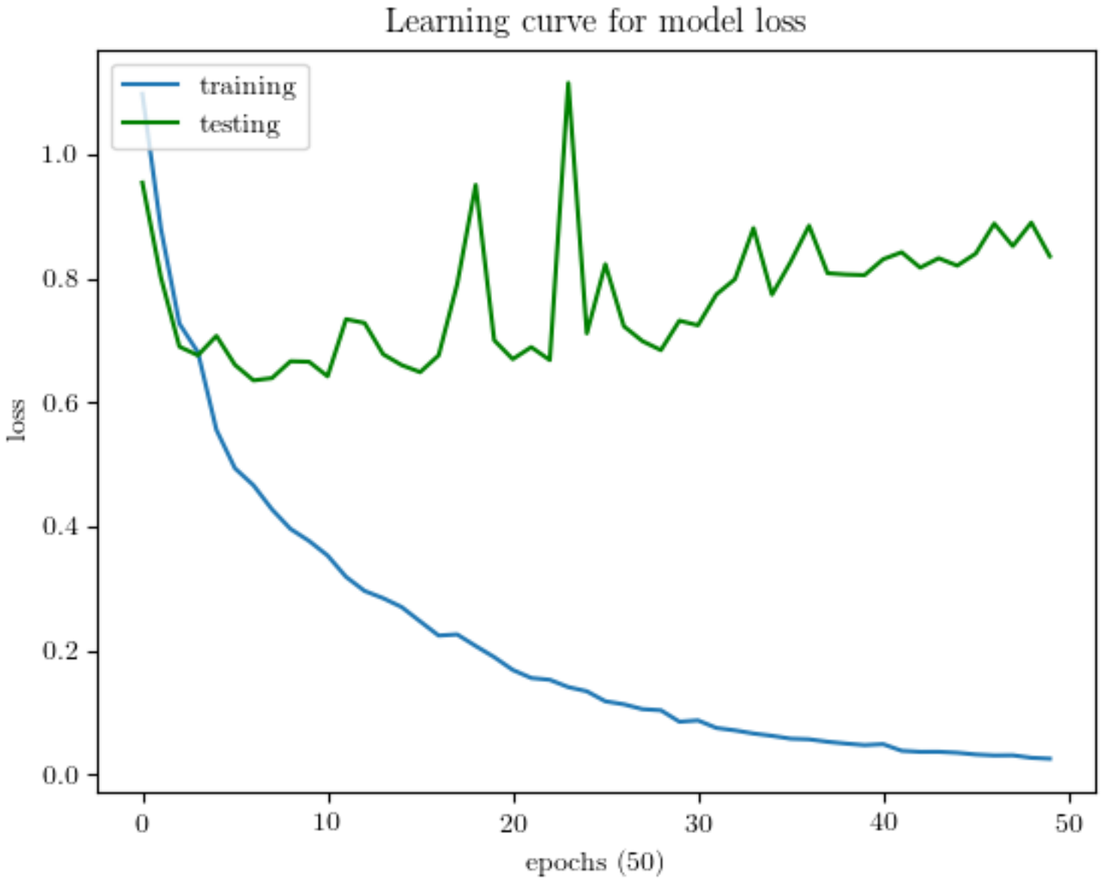
Results

- Split dataset into 80% training, 20% testing.
- Trained network for **50** epochs on one GPU (Nvidia).
- Tested 5 optimisers for 5 runs:
  - Adadelta, Adagrad, Adam, RMSprop & Stochastic Gradient Descent (SGD)
- Hyperparameters (Layer size, Max pooling size...)

- The network removes around 20-50% of neurons from the network when training. (Dropout)
  - Prevents overfitting the network.
- Rmsprop overfits the network.
- Adadgrad tends to underfit the network slightly.

- Adadelta & SGD optimiser works well with slight overfitting.
- Adam has comparable performance to SGD but underfits in some test runs.
  - Adadelta was the best, but SGD was better for 3 test runs. (Both achieved ~80% accuracy)


Learning curve for model accuracy
training
testing
accuracy
epochs (50)
1.0
0.9
0.8
0.7
0.6
0.5
0
10
20
30
40
50


Learning curve for model loss
training
testing
loss
epochs (50)
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50

~80% Model Accuracy

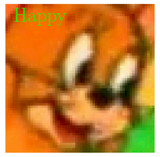
~90% Model Loss

Classification Results

Happy 86.8%
Surprise 9.5%
Angry 3.6%

Happy 97.9%
Surprise 1.5%
Angry 0.6%

Happy 91.0%
Surprise 5.5%
Angry 3.5%



Happy 52.7%
Angry 34.0%
Surprise 13.3%

Happy 55.8%
Surprise 33.2%
Angry 10.8%

Happy 67.7%
Angry 24.7%
Surprise 7.8%

Happy 50.5%
Angry 37.1%
Surprise 14.7%

Angry 48.5%
Happy 36.2%
Surprise 15.3%

Angry 55.7%
Happy 29.6%
Surprise 14.7%

Adadelta

Surprise 99.4%
Happy 0.5%
Angry 0.1%

Angry 100.0%
Happy 0.0%
Surprise 0.0%

Surprise 99.9%
Happy 0.1%
Angry 0.0%

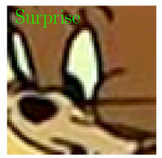
Happy 97.1%
Angry 2.8%
Surprise 0.1%

Happy 55.1%
Angry 44.7%
Surprise 0.2%

Angry 76.6%
Happy 23.4%
Surprise 0.0%

Surprise 99.3%
Angry 0.5%
Happy 0.2%

Happy 99.9%
Surprise 0.1%
Angry 0.0%

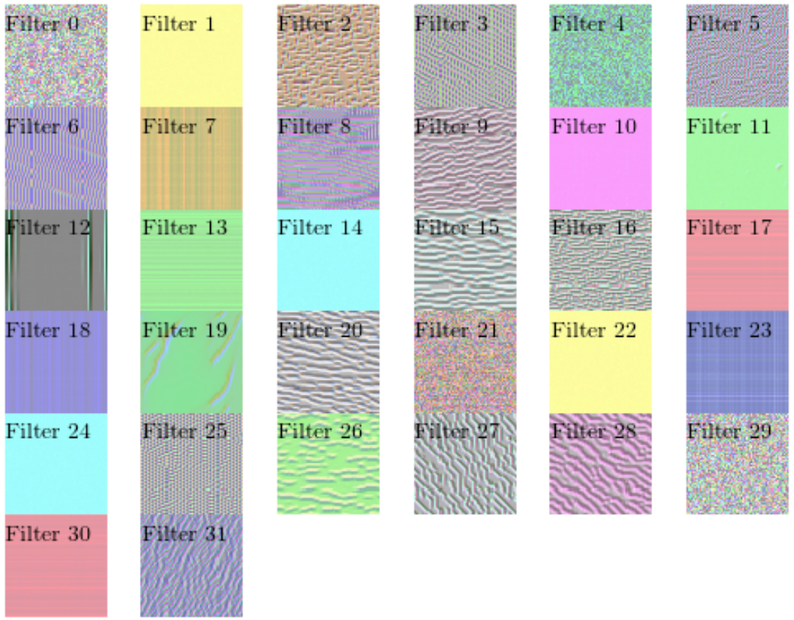


Surprise 88.3%
Angry 8.8%
Happy 2.9%

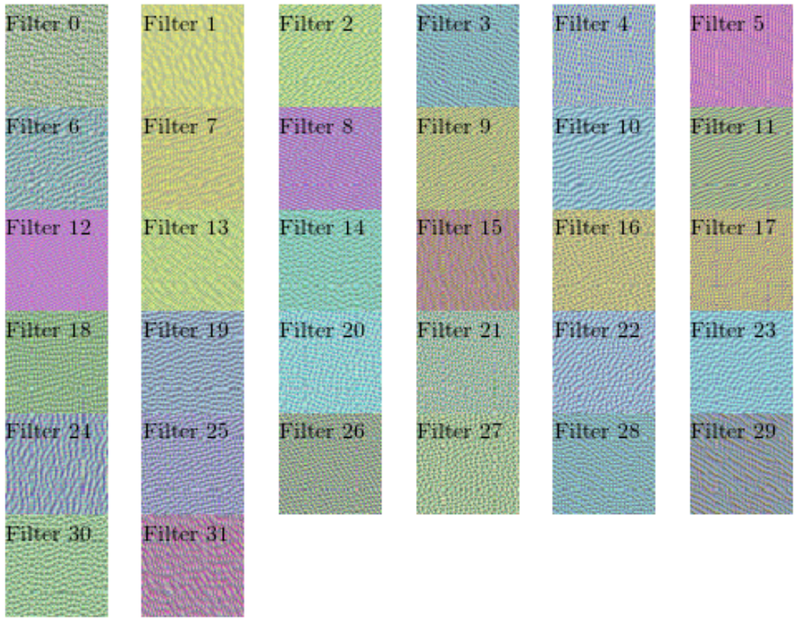
Stochastic Gradient Descent (SGD)

Convolution Visualisations

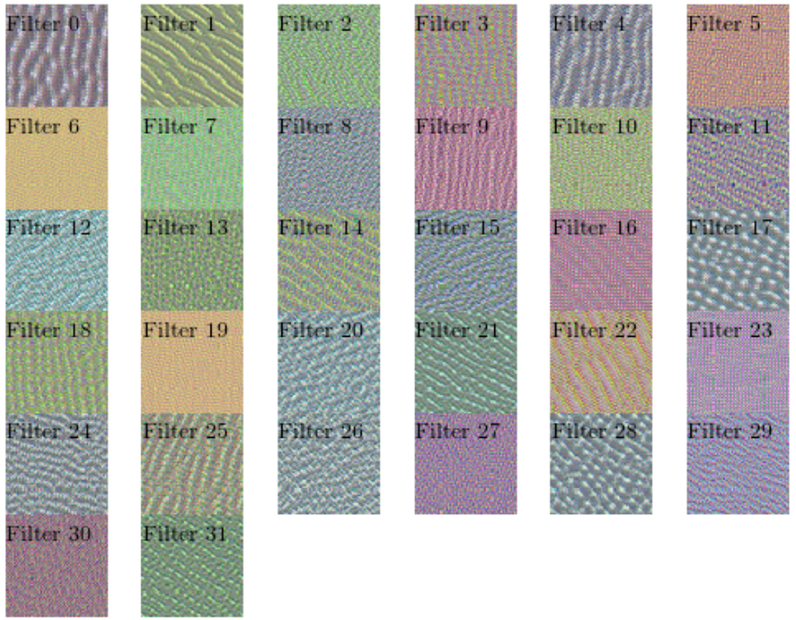
Visualisation of Convolution Layer 1



Visualisation of Convolution Layer 2



Visualisation of Convolution Layer 3



Potential Applications

- Animators
  - Automatic reference dataset.
  - Drawing -> Results of cartoons with similar emotions.
- Automatic subtitles.
- Recommendation Systems (Movies: which character is the happiest?)

Questions?

wesley@wesleyhill.co.uk
twitter.com/**@hakobyte**
github.com/**@hako**

github.com/hako/dissertation