Deliverable 3

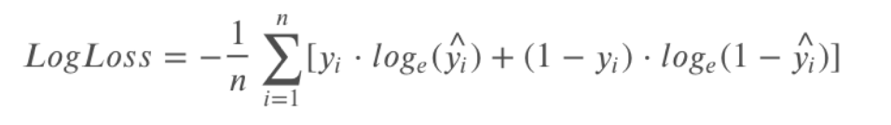
by Sarah Zhou

**Description**

Discussion of the final training results and the integration approach with a focus on the analysis of the results.

**1. Final Training Results**

As planned at the end of my previous deliverable, I decided to change and try other models for this deepfake challenge. I have tried to follow a notebook by Rob Mulla on Kaggle[[1]](#footnote-1) which uses SK learn and has successfully traced the faces in the videos. The loss function used is a log loss function and is as follows:



where:

* 𝑛 is the number of videos being predicted
* 𝑦*i* is the predicted probability of the video being FAKE
* 𝑦*i* is 1 if the video is FAKE, 0 if REAL
* 𝑙𝑜𝑔() is the natural (base e) logarithm

The distribution of the training set is as follows:

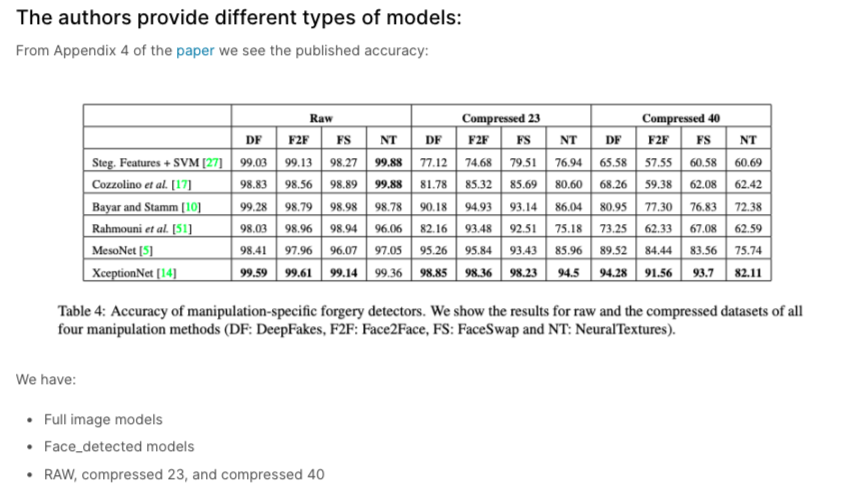
A screenshot of a cell phone

Description automatically generated

80.75% of the training videos are fake. I tested using the same distribution on the test set, as a result, predicting 0.80 scores worse than simply guessing 0.5, which means that this distribution is a failure.

I then proceed to explore a more complex pretrained models[[2]](#footnote-2) released by **FaceForensics++** in which they have used the **dlib c++ library[[3]](#footnote-3)**. It uses 3 types of image preprocessing and applied many models namely the Xception model, a 71-layer CNN pretrained on more than a million images from the ImageNet[[4]](#footnote-4).

The pretrained results are as below[[5]](#footnote-5):



The classification package[[6]](#footnote-6) and models[[7]](#footnote-7) are external. The comparison on training set resulted that **the Face\_detection Xception model with all compressed 23 images performed best.**

The predictions were run on 50 frames at a time and were averaged into a single value. 0.5 is the value returned when unable to predict. For every video, we return the maximum, minimum and average all "fake" prediction, which is all the frames compiled together[[8]](#footnote-8).

**Plot the Average vs Max Prediction Probability - Fake vs Real**

A screenshot of a cell phone

Description automatically generated

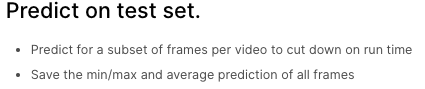
The trend is very faint, which means that the models is barely performing as desired.

In the **final training predictions**, the model has a somewhat clear binary partition, with almost no confusion on real videos.

A screenshot of a social media post

Description automatically generated

In the **final test predictions**, the model was confused about 80 videos.

 A screenshot of a social media post

Description automatically generated

**2. Final demonstration proposal**

I have decided to do a poster following the standard scientific method approach as my model does not have enough accuracy to be integrated in an app. I would present my learnings from the project and models.

Problem and hypothesis:

* Tackle the extremely difficult problem to detect deepfake in videos as a current issue in the online community.

Concrete results:

* With pretrained models of **FaceForensics++**, 80 videos of the test set was not able to be predicted. The rest of the videos follows a somewhat normal distribution. (Since Kaggle holds a private test set, we cannot compare thoroughly the results. e.g. See how the test set differs from the training set to then conduct an analysis.)
* The baseline implementations were various and vague submitted by participants for the purpose of the competition. I have saved all papers and previous work consulted. I will be able to also explain the different models I have explored for this challenge.
* This challenge certainly holds high value in terms of application in real world products. A successful model can be commercialized to help the public detect fake news, help in cyber security, assist the police for criminal cases and serve as an educational tool.

Key ideas:

* Focus on the learning and different approaches possible to try to solve this challenge.
* Show graphs and failed models and their unsatisfactory learning trend.
* Using visual prints of frames of some sample real and fake videos, explain the difficulties of deepfake detection by showing the similarities of human faces and deepfake faces, but also the caveats (e.g. unnatural blinking patterns) between the two that we can use and explore further upon.

**3. References**

A screenshot of a cell phone

Description automatically generated

Davis E. King. Dlib-ml: A Machine Learning Toolkit.

Journal of Machine Learning Research, 2009

@Article{dlib09,

author = {Davis E. King},

title = {Dlib-ml: A Machine Learning Toolkit},

journal = {Journal of Machine Learning Research},

year = {2009},

volume = {10},

pages = {1755-1758},

}

1. <https://www.kaggle.com/robikscube/kaggle-deepfake-detection-introduction> [↑](#footnote-ref-1)
2. <https://www.kaggle.com/robikscube/faceforensics-baseline-dlib-no-internet#Running-FaceForensics++-in-a-Kaggle-Notebook> [↑](#footnote-ref-2)
3. <http://dlib.net/ml.html> [↑](#footnote-ref-3)
4. <https://www.mathworks.com/help/deeplearning/ref/xception.html> [↑](#footnote-ref-4)
5. <https://arxiv.org/pdf/1901.08971.pdf> [↑](#footnote-ref-5)
6. <https://github.com/ondyari/FaceForensics/tree/master/classification#classification> [↑](#footnote-ref-6)
7. <http://kaldir.vc.in.tum.de/FaceForensics/models/faceforensics++_models.zip> [↑](#footnote-ref-7)
8. <https://www.kaggle.com/robikscube/faceforensics-baseline-dlib-no-internet#Running-FaceForensics++-in-a-Kaggle-Notebook> [↑](#footnote-ref-8)