Evaluating the Resiliency of Blink-Based DeepFake Detection Against Adversarial Noise

By Joel Coulon (u2204489)

A Brief Introduction to DeepFakes

- A DeepFake is where a piece of media (usually images and videos) are digitally altered or created by an AI
- Whilst originally created for entertainment purposes
- Can be used for misinformation, scams, and various other nefarious activities
- They are frighteningly realistic:

Image from: https://www.whichfaceisreal.com/

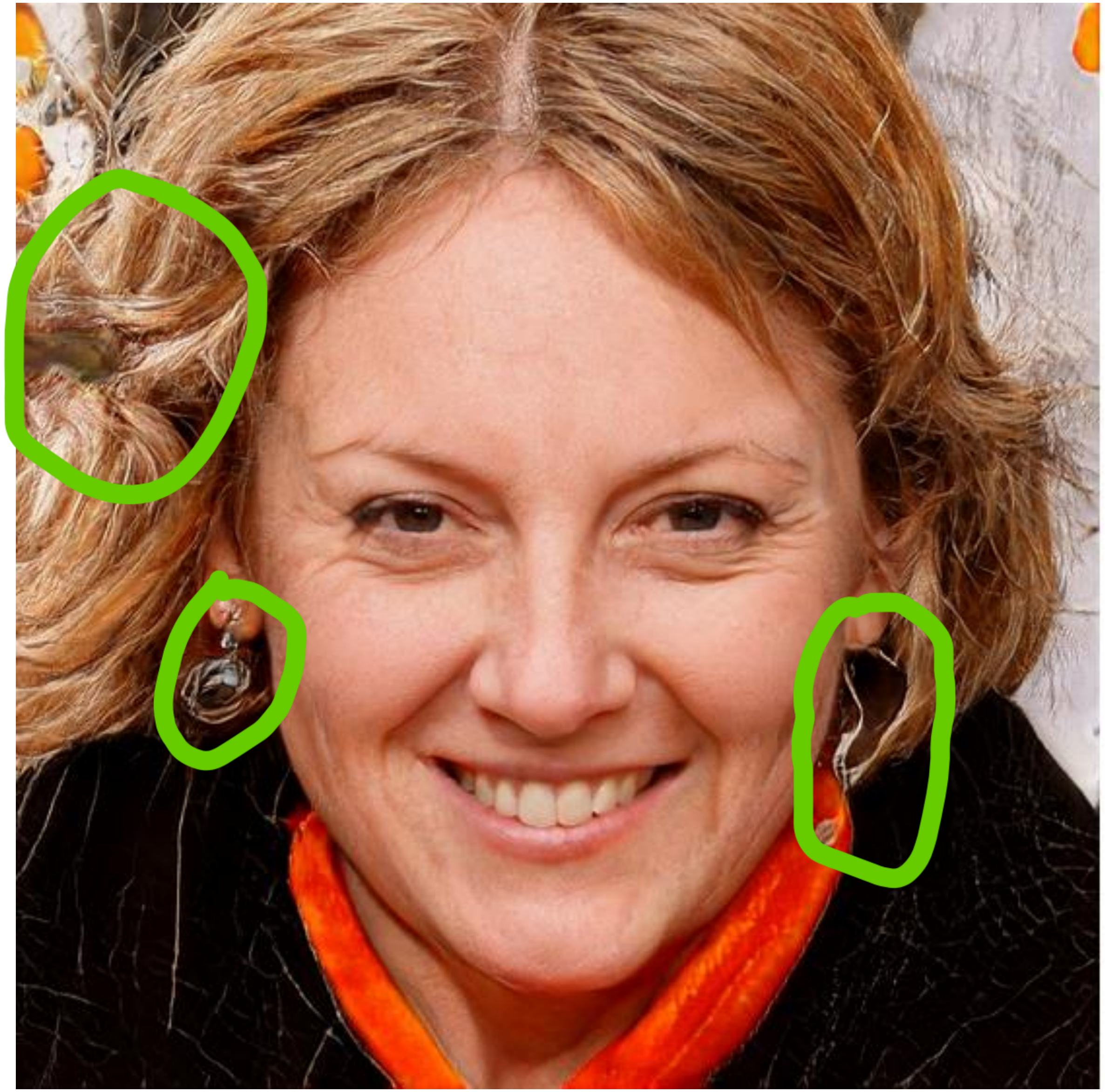


Image from: https://www.whichfaceisreal.com/results.php?r=1&p=1&i1=image-2019-02-17_004111.jpeg&i2=68128.jpeg

Motivation

- Wanted to do a project related to cybersecurity and AI
- A friend suggested looking into DeepFakes
- Countering Malicious DeepFakes: Survey, Battleground, and Horizon
  - “[DeepFakes Detectors are] vulnerable to adversarial noise attacks with imperceptible additive noises”
  - “[DeepFakes] do not take physiological signals such as eye blink frequency … into consideration”

Juefei-Xu, Felix, Run Wang, Yihao Huang, Qing Guo, Lei Ma, and Yang Liu. “Countering malicious deepfakes: Survey, battleground, and horizon”. *International journal of computer vision* 130, no. 7 (2022): 1678-1734.

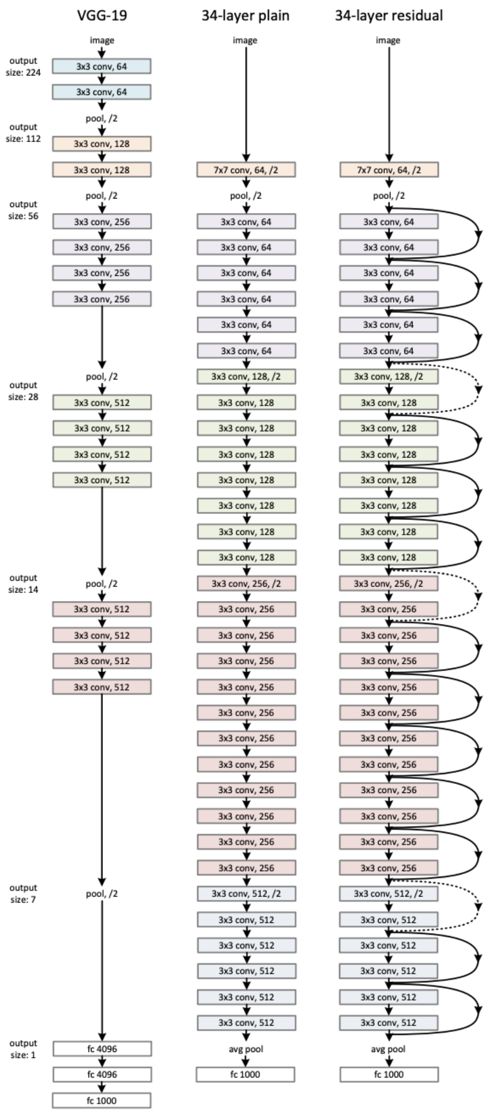
Adversarial Noise

Causing traditional DeepFake detectors to misclassify fake images

Traditional DeepFake Detectors

- Use backbones
  - Pretrained convolutional neural networks
  - Based on existing architectures (for example ResNet)
- Binary classifier added to the head to fine-tune

```
resnet = ResNet50(weights="imagenet", include_top=False, input_shape=input_shape)
model = Sequential()
model.add(resnet)
model.add(GlobalAveragePooling2D())
model.add(Dense(64, activation="relu"))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(2, activation="softmax"))
```

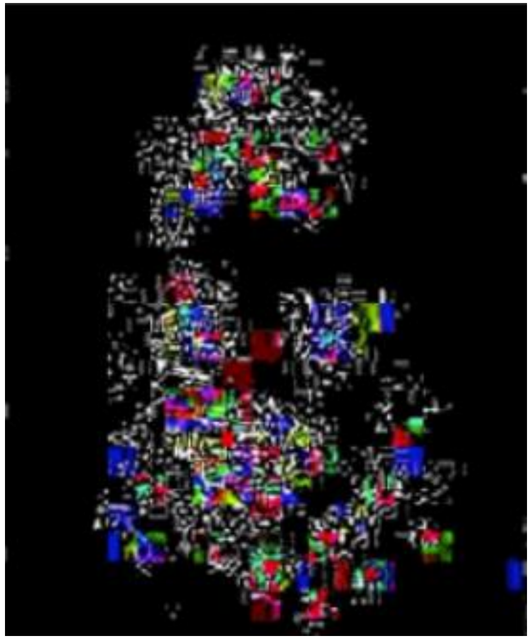


He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. 2016.

Adversarial Noise

Classified as fake

+



=

Classified as real

Gandhi, Apurva, and Shomik Jain. "Adversarial perturbations fool deepfake detectors". In *2020 International joint conference on neural networks (IJCNN)*, pp. 1-8. IEEE, 2020.

Adversarial Noise

CW-L2 attack

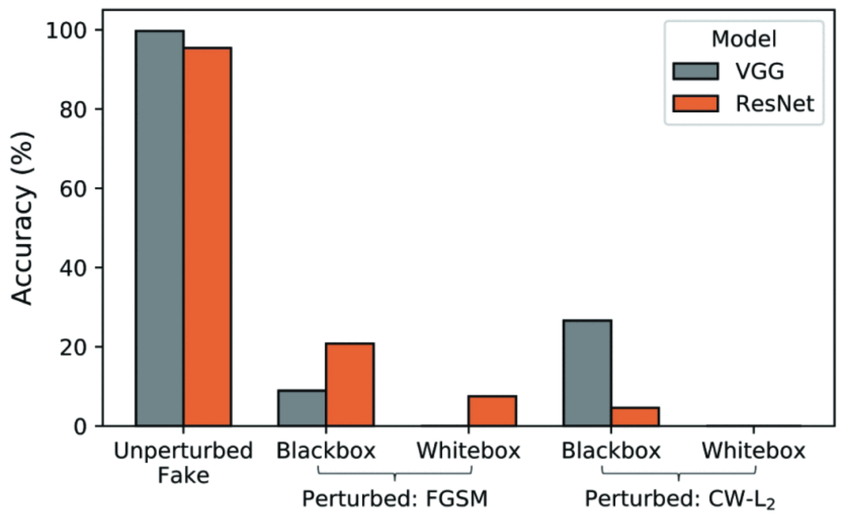
- Adds noise that minimises L2 norm of the noise (keeps close to original image)
- But still causes a misclassification

$$\mathbf{x}_{adv} = \frac{1}{2}(\tanh(\omega^*) + 1)$$
$$\omega^* = \arg\min_\omega \left\{ \|\mathbf{x}' - \mathbf{x}\|_2^2 + cf(\mathbf{x}') \right\}$$
$$f(\mathbf{x}') = \max\left(\max_{i \neq y}\left\{\mathbf{Z}(\mathbf{x}')_y - \mathbf{Z}(\mathbf{x}')_i\right\}, -\kappa\right)$$

FGSM

- Finds gradient of model's loss function, adds a small amount of noise to that gradient to cause the model to misclassify

$$\mathbf{x}_{adv} = \mathbf{x} + \varepsilon\,\mathrm{sign}(\nabla_x J(\mathbf{x}, \mathbf{y}, \theta)).$$



Gandhi, Apurva, and Shomik Jain. "Adversarial perturbations fool deepfake detectors". In *2020 International joint conference on neural networks (IJCNN)*, pp. 1-8. IEEE, 2020.

Adversarial Noise (cont.)

- **FakeRetouch**
  - Add gaussian noise to an image

$$\hat{\mathbf{I}} = \mathbf{K} \circledast (\mathbf{I} + \mathbf{A} \odot \mathbf{N}_\sigma) \qquad \arg\max_{\mathbf{A}} J(\mathbf{D}(\mathbf{I} + \mathbf{A}), y) + \|\mathbf{A}\|_1$$

  - Where based on based on binary map **A**
  - Creates Kernel **K** using a neural network
  - Compute noise-map **A** by minimising L1 loss

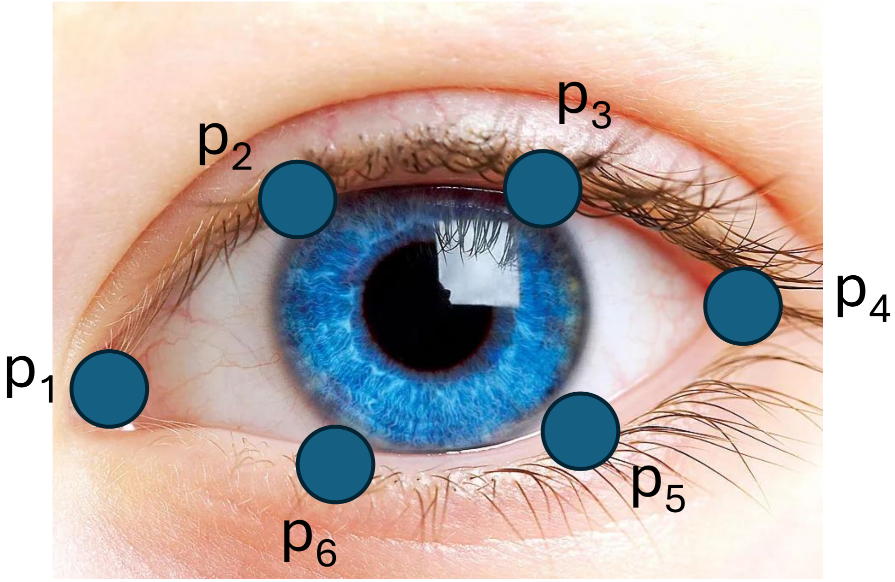
	Accuracy(%)
Fake	88.99
FR(rn)-gau	22.59 (-66.4)
FR(rn)-uni	21.73 (-67.26)
FR(an)-uni	21.64 (-67.35)

Huang, Yihao, Felix Juefei-Xu, Qing Guo, Xiaofei Xie, Lei Ma, Weikai Miao, Yang Liu, and Geguang Pu. "Fakeretouch: Evading deepfakes detection via the guidance of deliberate noise". *arXiv preprint arXiv:2009.09213* 1, no. 2 (2020).

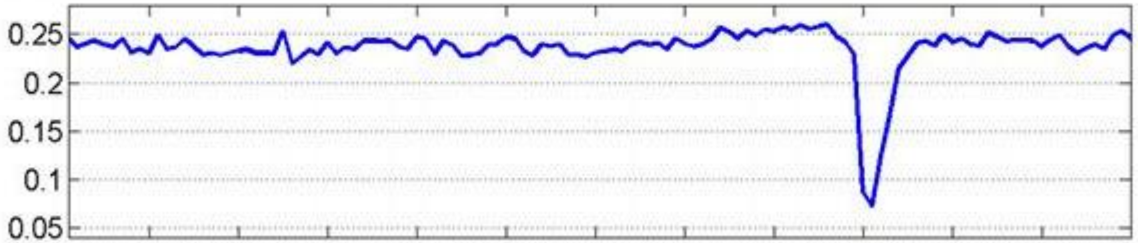
Blinking

Detecting DeepFakes via blinking inconsistencies

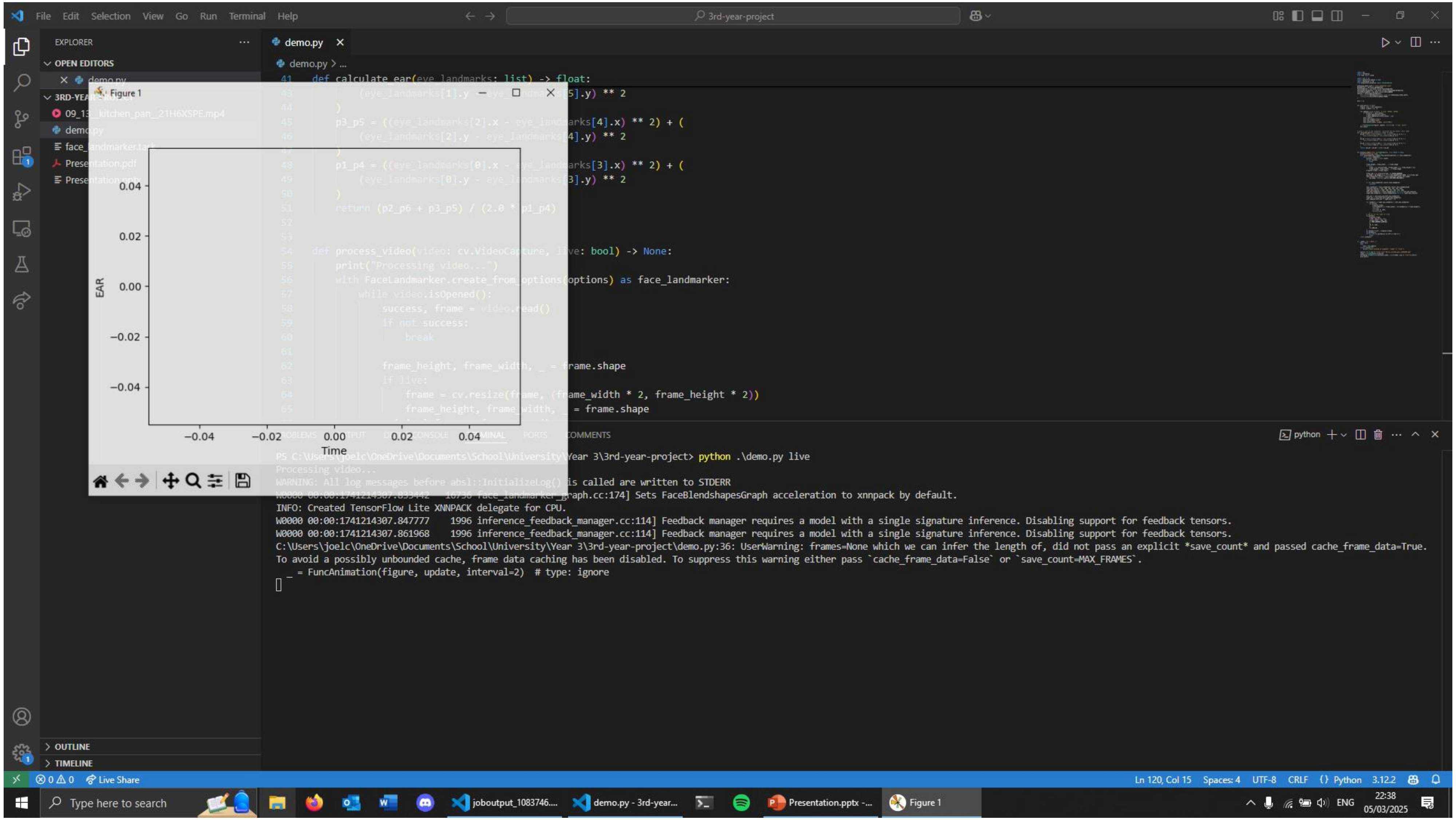
Eye Aspect Ratio (EAR)



$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$



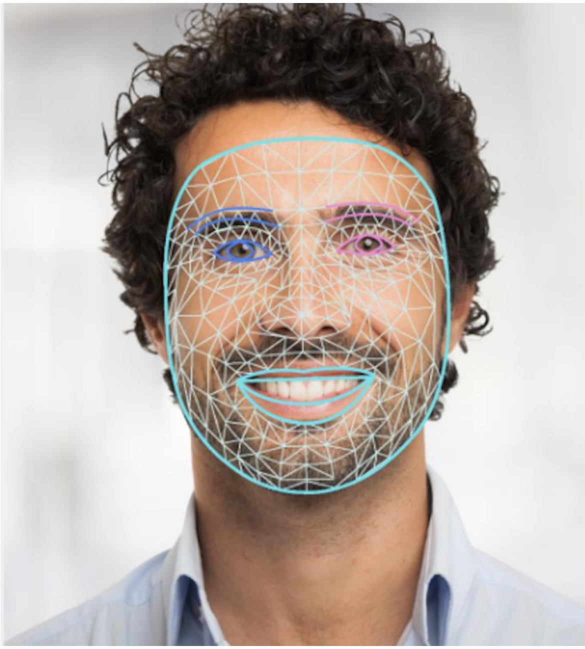
Soukupova, Tereza, and Jan Cech. "Eye blink detection using facial landmarks." In *21st computer vision winter workshop, Rimske Toplice, Slovenia*, vol. 2, p. 4. 2016.


File Edit Selection View Go Run Terminal Help
3rd-year-project
EXPLORER
OPEN EDITORS
demo.py
Figure 1
EAR
Time
0.04
0.02
0.00
−0.02
−0.04
−0.04 −0.02 0.00 0.02 0.04
def calculate_ear(eye_landmarks: list) -> float:
p3_p5 = ((eye_landmarks[2].x - eye_landmarks[4].x) ** 2) + (
p1_p4 = ((eye_landmarks[0].x - eye_landmarks[3].x) ** 2) + (
return (p2_p6 + p3_p5) / (2.0 * p1_p4)
def process_video(video: cv.VideoCapture, live: bool) -> None:
print("Processing video...")
with FaceLandmarker.create_from_options(options) as face_landmarker:
frame_height, frame_width, _ = frame.shape
frame = cv.resize(frame, (frame_width * 2, frame_height * 2))
frame_height, frame_width, _ = frame.shape
PS C:\Users\joelc\OneDrive\Documents\School\University\Year 3\3rd-year-project> python .\demo.py live
Processing video...
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
W0000 00:00:1741214307.847777 1996 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.
W0000 00:00:1741214307.861968 1996 inference_feedback_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.
C:\Users\joelc\OneDrive\Documents\School\University\Year 3\3rd-year-project\demo.py:36: UserWarning: frames=None which we can infer the length of, did not pass an explicit *save_count* and passed cache_frame_data=True. To avoid a possibly unbounded cache, frame data caching has been disabled. To suppress this warning either pass `cache_frame_data=False` or `save_count=MAX_FRAMES`.
_ = FuncAnimation(figure, update, interval=2) # type: ignore
OUTLINE
TIMELINE
Live Share
Ln 120, Col 15 Spaces: 4 UTF-8 CRLF Python 3.12.2
Type here to search
demo.py - 3rd-year...
Presentation.pptx -...
Figure 1
ENG
22:38
05/03/2025

Proof of Concept

Is my theory correct?

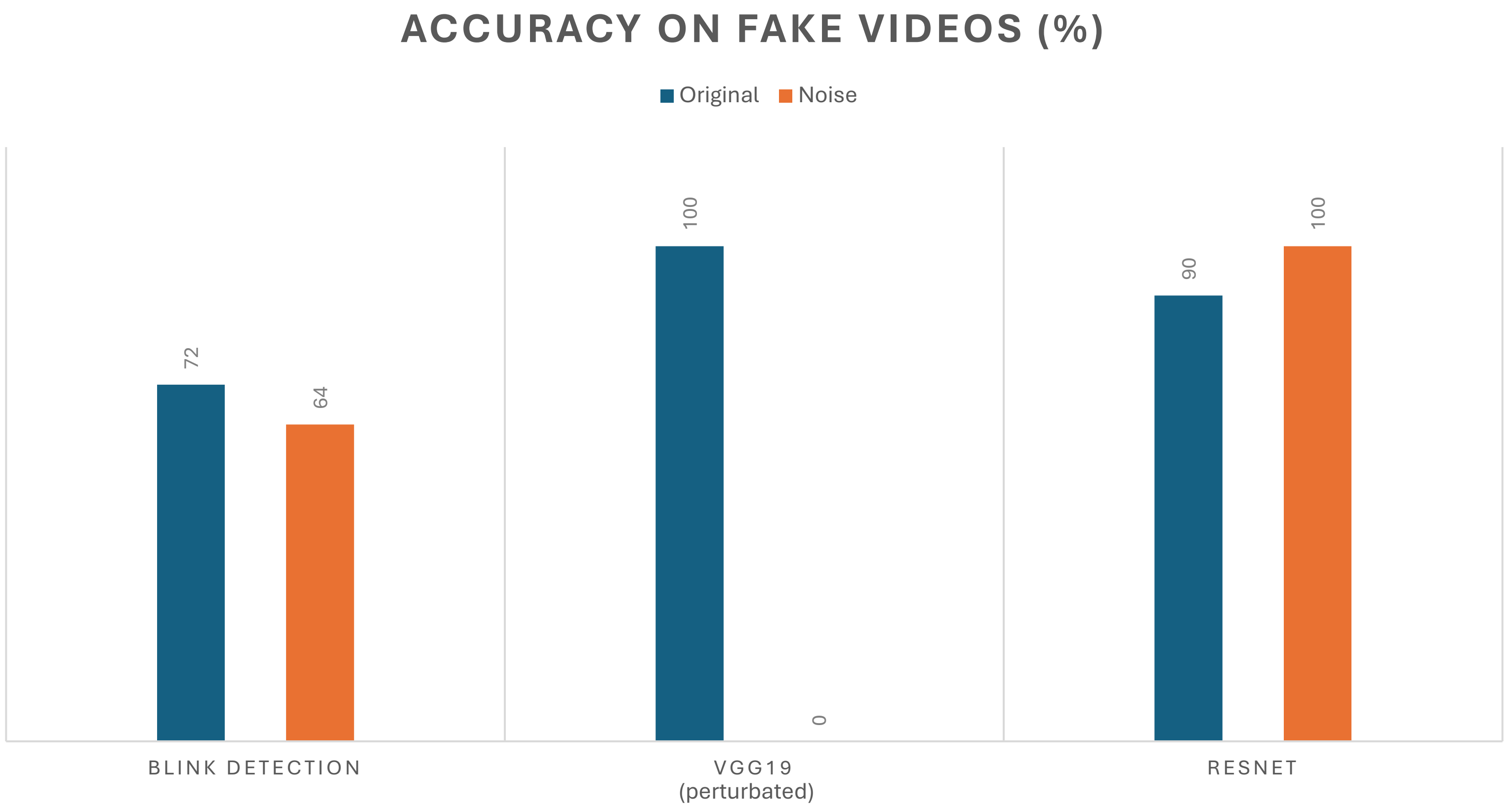
Proof of Concept



- Made over the Christmas holidays
- Uses pre-existing methods where possible
- Google's MediaPipe[1] for eye landmarks
- Compare number of blinks compared to the human average
- Traditional detectors represented by VGG19 and ResNet detectors
- FGSM noise using Foolbox[2,3]
  - Noise targeting VGG19

[1] Lugaresi, Camillo, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang et al. "Mediapipe: A framework for building perception pipelines". *arXiv preprint arXiv:1906.08172* (2019).
[2] Rauber, Jonas, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. "Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax". *Journal of Open Source Software* 5, no. 53 (2020): 2607.
[3] Rauber, Jonas, Wieland Brendel, and Matthias Bethge. "Foolbox v0. 8.0: A python toolbox to benchmark the robustness of machine learning models". *CoRR* (2017).

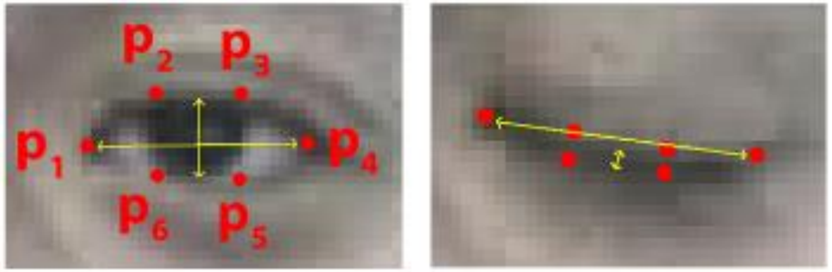
Results of Proof of Concept

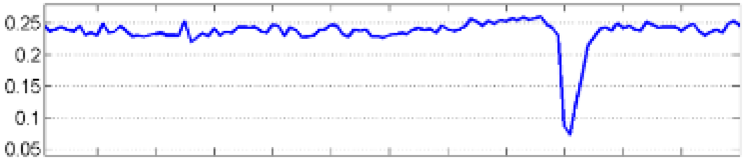


- Noise very specialised to each model
- When varying ε, ResNet would change, Blink Detection would not

The final model

Proposed Architecture for Detection





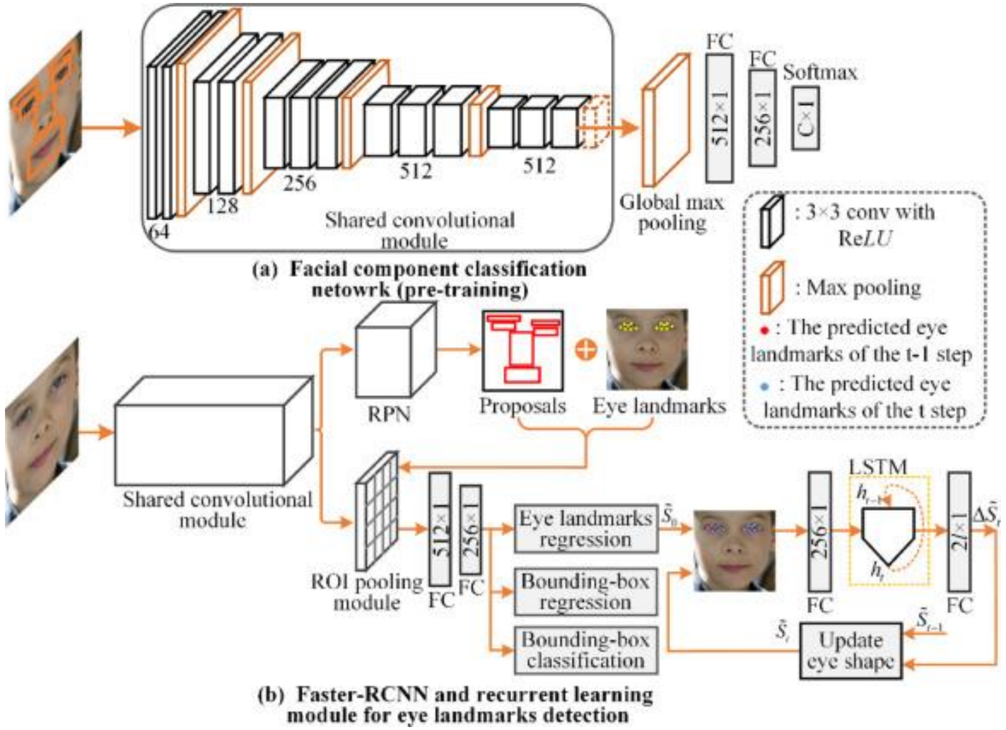
EAR Analysis

Selection of Eye Landmark Model

- The vast majority of public facial landmarking models are unsuitable
- They are either optimised for model size or model speed
- Not model accuracy

Facial landmark detection by semi-supervised deep learning

- Most accurate eye landmark detector currently published
- RPN blackbox?
- FC Layer?
- So many bugs…
- Scrapped development after a month of work



Huang, Bin, Renwen Chen, Qinbang Zhou, and Wang Xu. “Eye landmarks detection via weakly supervised learning”. *Pattern Recognition* 98 (2020): 107076.

Papers With Code

Search | Browse State-of-the-Art | Datasets | Methods | More | Sign In

Computer Vision

Facial Landmark Detection

Edit

47 papers with code • 10 benchmarks • 16 datasets

Facial Landmark Detection is a computer vision task that involves detecting and localizing specific points or landmarks on a face, such as the eyes, nose, mouth, and chin. The goal is to accurately identify these landmarks in images or videos of faces in real-time and use them for various applications, such as face recognition, facial expression analysis, and head pose estimation.

(Image credit: Style Aggregated Network for Facial Landmark Detection)

Benchmarks

Add a Result

These leaderboards are used to track progress in Facial Landmark Detection

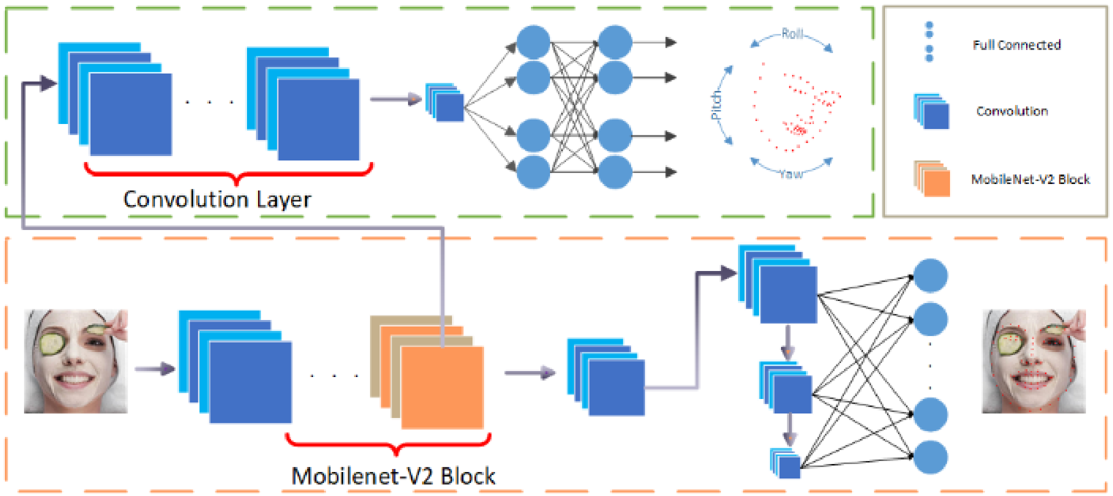
Trend	Dataset	Best Model	Paper	Code	Compare
	300W	D-ViT			See all
	AFLW-Full	FiFA			See all
	AFLW-Front	FiFA			See all
	WFLW	D-ViT			See all
	CatFLW	ELD (EfficientNetV2S)			See all

Content

New Detection Methods

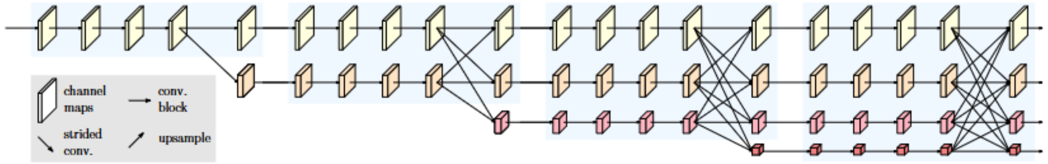
PFLD[1] (yet to be implemented)

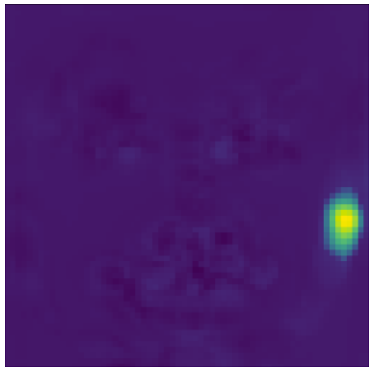
- Uses pre-trained backbone network to do initial predictions
    - Currently uses MobileNetV2 (interchangeable?)
- A second model picks up from an intermediary layer to estimate pitch, yaw, and roll
- Used in loss function

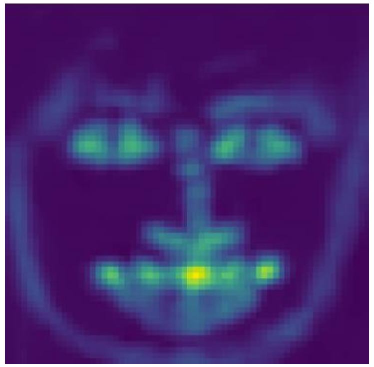


HRNet[2]

- Multiple modular high to low fusion blocks in parallel
- Lower blocks are downsampled to focus on finer features
- Output is a heatmap per landmark, max value of heatmap is landmark

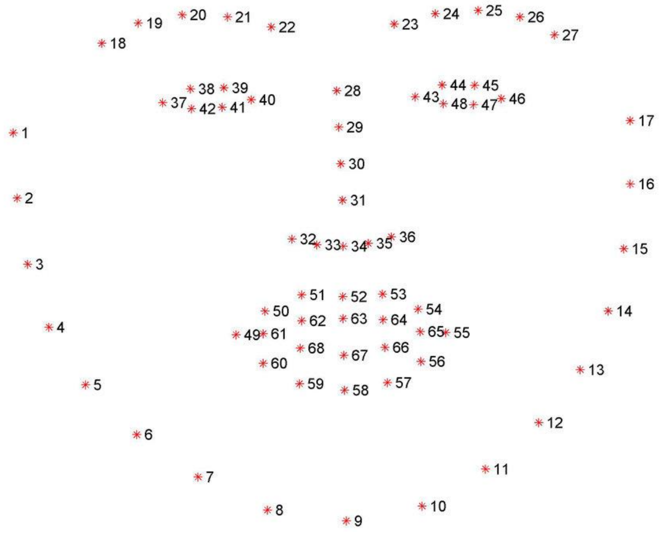






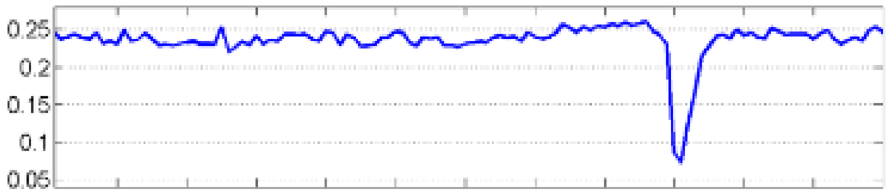
[1] Guo, Xiaojie, Siyuan Li, Jinke Yu, Jiawan Zhang, Jiayi Ma, Lin Ma, Wei Liu, and Haibin Ling. “PFLD: A practical facial landmark detector”. *arXiv preprint arXiv:1902.10859* (2019).
[2] Sun, Ke, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. “High-resolution representations for labeling pixels and regions”. *arXiv preprint arXiv:1904.04514* (2019).

Datasets – Facial Landmarks



- Various datasets exist for facial landmarking
- Vast majority use 68-landmarks
  - Subsampled 68 if necessary
- 7 Datasets used (46,000 images)
- Chosen for wide variety of facial poses, situations, and occlusions

EAR Analysis



- Now have an EAR-time graph
- Can be abstracted to univariate time series
- A variety of analysis methods exist, both classical and neural-network-based
  - Time Series Classification: A Review of Algorithms and Implementations[1]
  - LSTM Fully Convolutional Networks for Time Series Classification[2]
  - Deep Learning for Time Series Classification: A Review[3]

[1] Faouzi, Johann. "Time series classification: A review of algorithms and implementations". *Machine Learning (Emerging Trends and Applications)* (2022).
[2] Karim, Fazle, Somshubra Majumdar, Houshang Darabi, and Shun Chen. "LSTM fully convolutional networks for time series classification". *IEEE access* 6 (2017): 1662-1669.
[3] Ismail Fawaz, Hassan, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. "Deep learning for time series classification: a review". *Data mining and knowledge discovery* 33, no. 4 (2019): 917-963.

Methods Evaluated

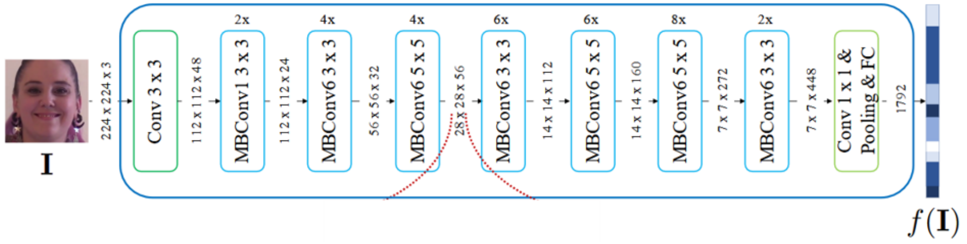
- 12 methods were analysed
  - 5 neural-network-based methods
  - 7 traditional methods
- Fully Convolutional Neural Network
  - 79% effective

```
x = Conv1D(128, 8, padding="same")(input)
x = BatchNormalization()(x)
x = Activation("relu")(x)

x = Conv1D(256, 5, padding="same")(x)
x = BatchNormalization()(x)
x = Activation("relu")(x)

x = Conv1D(128, 3, padding="same")(x)
x = BatchNormalization()(x)
x = Activation("relu")(x)

x = GlobalAveragePooling1D()(x)
x = Dense(2, activation="softmax")(x)
```

Noise & DeepFake Detectors

- The same CW-L2, FGSM, and FakeRetouch were used

- DeepFake Detectors
  - XceptionNet[1]
  - EfficientNetB4[2]



	FaceForensics++	EfficientNetB4 + EfficientNetB4ST + B4Att + B4AttST
	FaceForensics	XceptionNet

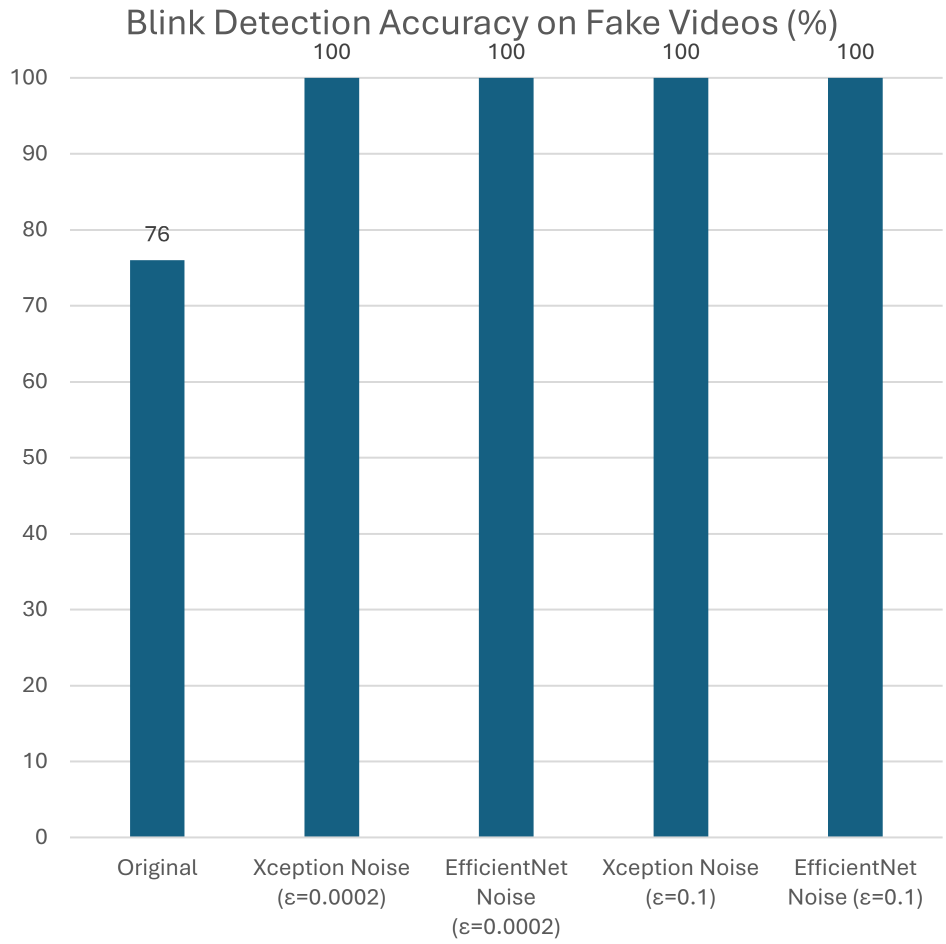
[1] Rossler, Andreas, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. "Faceforensics++: Learning to detect manipulated facial images". In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1-11. 2019.
[2] Bonettini, Nicolo, Edoardo Daniele Cannas, Sara Mandelli, Luca Bondi, Paolo Bestagini, and Stefano Tubaro. "Video face manipulation detection through ensemble of cnns". In *2020 25th international conference on pattern recognition (ICPR)*, pp. 5012-5019. IEEE, 2021.

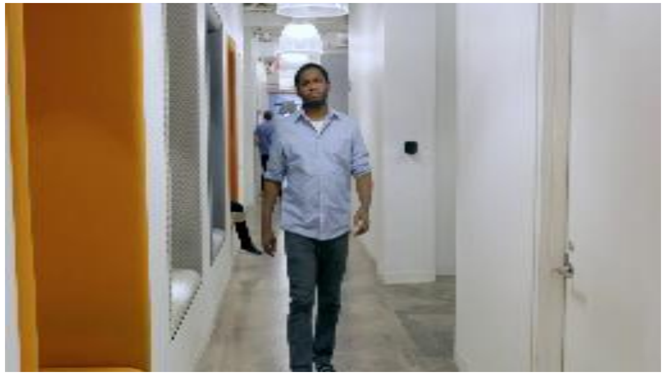
Datasets - DeepFakes

- FaceForensics++
  - A subset of 100 videos for training (50 real, 50 fake)
  - A subset of 100 videos for testing (50 real, 50 fake)
  - The entire dataset to be used for report
- More to be added in final report (FakeAVCeleb, DFDC,...)

Rossler, Andreas, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. "Faceforensics++: Learning to detect manipulated facial images." In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1-11. 2019.

Results


Blink Detection Accuracy on Fake Videos (%)
100
90
80
70
60
50
40
30
20
10
0
76
100
100
100
100
Original
Xception Noise (ε=0.0002)
EfficientNet Noise (ε=0.0002)
Xception Noise (ε=0.1)
EfficientNet Noise (ε=0.1)

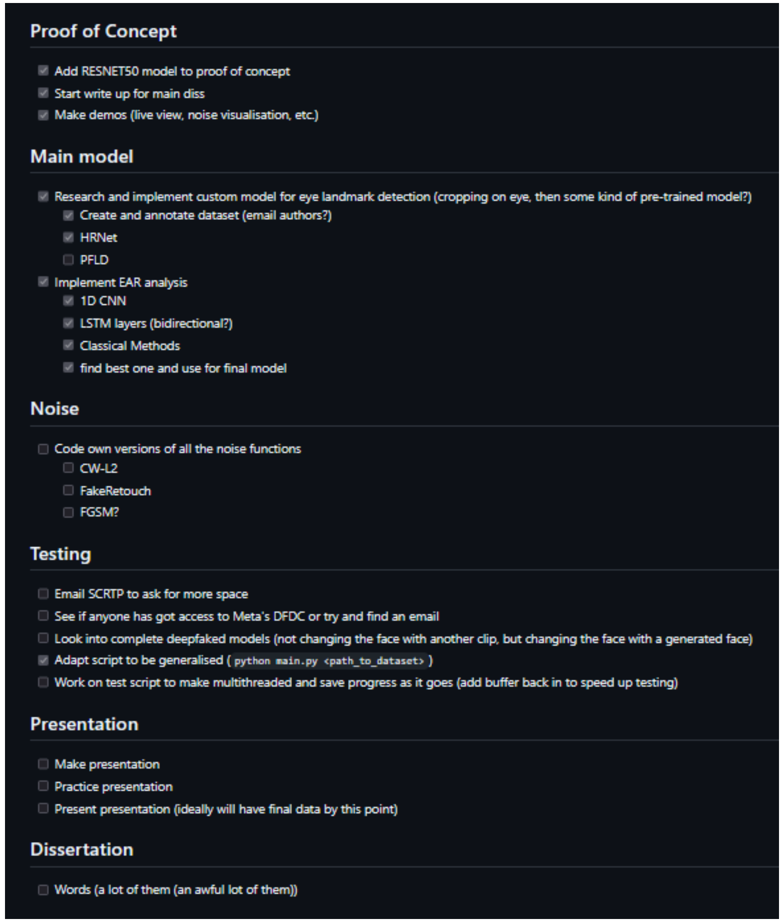


Why?

- Dealing with the video over time
- Noise needs to be consistent over time
- This is not possible with current noise methodologies

Project Management

- Progress tracked in a central document
- Timeline via Gantt Chart
  - Buffer weeks were used
- Switched to eye landmark models with pre-existing implementations
- Analysis was sped up thanks to pre-existing libraries



Proof of Concept

- [x] Add RESNET50 model to proof of concept
- [x] Start write up for main diss
- [x] Make demos (live view, noise visualisation, etc.)

Main model

- [x] Research and implement custom model for eye landmark detection (cropping on eye, then some kind of pre-trained model?)
  - [x] Create and annotate dataset (email authors?)
  - [x] HRNet
  - [ ] PFLD
- [x] Implement EAR analysis
  - [x] 1D CNN
  - [x] LSTM layers (bidirectional?)
  - [x] Classical Methods
  - [x] find best one and use for final model

Noise

- [ ] Code own versions of all the noise functions
  - [ ] CW-L2
  - [ ] FakeRetouch
  - [ ] FGSM?

Testing

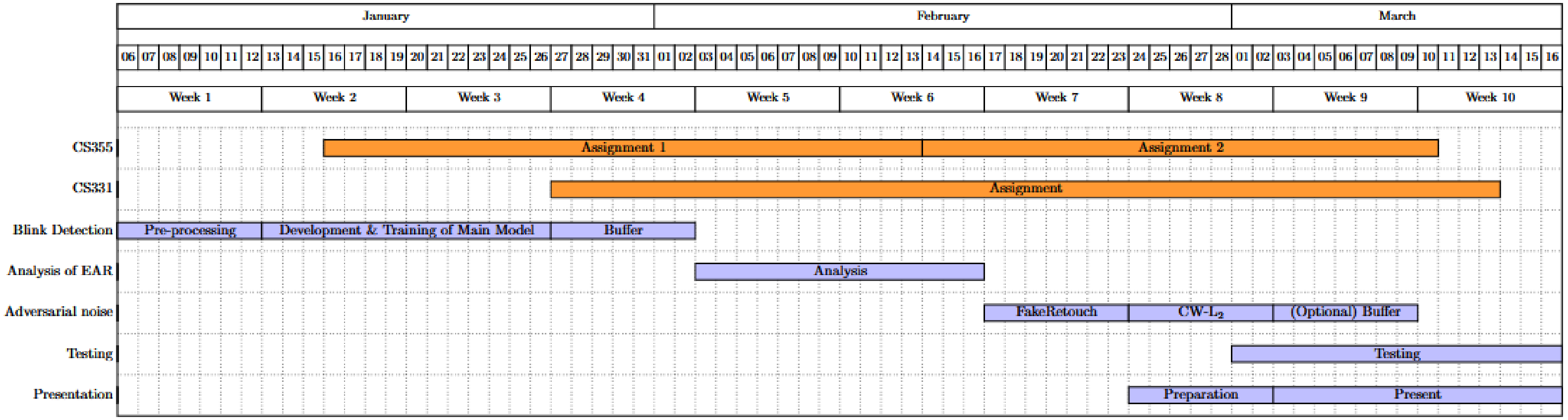
- [ ] Email SCRTP to ask for more space
- [ ] See if anyone has got access to Meta's DFDC or try and find an email
- [ ] Look into complete deepfaked models (not changing the face with another clip, but changing the face with a generated face)
- [x] Adapt script to be generalised (`python main.py <path_to_dataset>`)
- [ ] Work on test script to make multithreaded and save progress as it goes (add buffer back in to speed up testing)

Presentation

- [ ] Make presentation
- [ ] Practice presentation
- [ ] Present presentation (ideally will have final data by this point)

Dissertation

- [ ] Words (a lot of them (an awful lot of them))



Still To Do

- Test on a wide variety of datasets
- Evaluate transferability
- Implement PFLD and test
- Implement FakeRetouch
- Future Research
  - Development of time-sensitive noise
  - Diffusion model to reduce noise
  - Other temporal dependencies (breathing, heartbeat)

Accuracy (%) of fine-tuned ResNet

Tested on (columns); *Trained on* (rows)

Data Set	Celeb DF v1	Stylegan2	Stylegan3-t	Stylegan3-r	DFDC Pt. 0
Celeb DF v1	99.1	44.2	44.2	44.0	51.2
Stylegan2	24.1	98.7	52.9	48.4	57.4
Stylegan3-t	16.7	69.7	96.7	84.0	7.0
Stylegan3-r	16.9	68.0	89.0	97.2	7.0
DFDC Pt. 0	68.1	57.4	57.5	57.5	88.7

Gallagher, Shannon K. "Machine Learning for Deepfake Detection." *Carnegie Mellon University. 2022*

Thanks for listening!