Adaptive Modelling for Deepfake Detection using Post-Training

1. Introduction

Video forgery technology like Deepfake can be a threat to society if used by people with malicious intent. State-of-the-art forgery techniques are so real that even human eyes could not detect.

In the other part of the project, we proposed an adaptive modelling technique for multi-class scenario. In this part of the project, we adapted the same technique to deal with 2-class problem. Specifically, given a deepfake detection model trained without using adversarial attack samples, we apply the same adaptive modelling (post-training) concept to this deepfake detection model to guard against adversarial attacks. We assume that we already have a deepfake detection model, and we have at our disposal the dataset for post-training and testing. As in the multiclass problem, the intuition in our project is that given any natural input data point, its corresponding adversarial data point should be near a decision boundary. We implemented two image embedding search algorithms to find images in training set that are close to the test image. More details can be found in Algorithm1 and Algorithm2. The original model is fine-tuned with this small amount of data found so as to give an accurate prediction that is more robust against potential adversarial attacks. Experiment results show that without retraining and introducing additional data, our technique can improve the robust accuracy for Deepfake detection models that are vulnerable to adversarial attacks.

2. Related Work

Madry et al. showed that when using the fast-single iteration method (FGSM), the trained model is still vulnerable to stronger multi-iteration adversarial samples. They(Madry et al., 2017) suggested using a multi- iteration method called projected gradient descent (PGD) to train the model so that the model would have universal robustness against adversarial samples. Seyed et al. (Moosavi-Dezfooli et al., 2016)proposed the DeepFool algorithm. DeepFool can generate adversarial examples with minimum perturbation to change the classifier's decision. We note that the adversarial samples generated by Deepfool are very close to the classification boundary.

(Rossler et al., 2019) proposed FaceForensics++ Dataset using Face2Face, FaceSwap, DeepFakes and NeuralTextures four automatic face forgery algorithms to process 1,000 real video sequences. The data came from 977 YouTube videos, all of which contain trackable and mostly unoccluded frontal faces. The paper trained the XceptionNet(Chollet, 2017) classification network to distinguish all four manipulation methods.

3. Methods

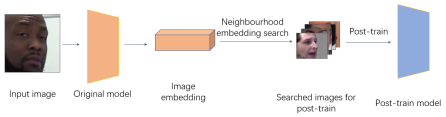
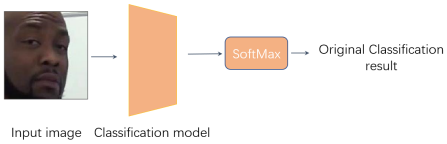
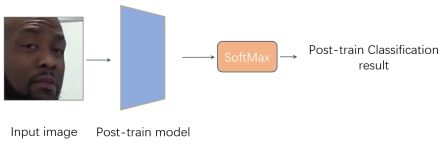


Figure 1. Overview of Adaptive Modelling using Post-Training. The proposed architecture consists of three steps at inference stage: getting image embedding, neighbourhood embedding search and fine-tuning model.



(a) Inference stage of original classification model.



(b) Inference stage of classification model fine-tuned by proposed method.

Figure 2. Inference stage of original classification model and using proposed post-train method. Different color reveals that model parameters are different.

We describe below how to get image embedding; how to implement neighbourhood embedding search algorithm; and how to fine-tune the model.

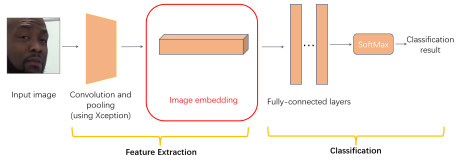


Figure 3. Process of getting image embedding from a trained Xception model.

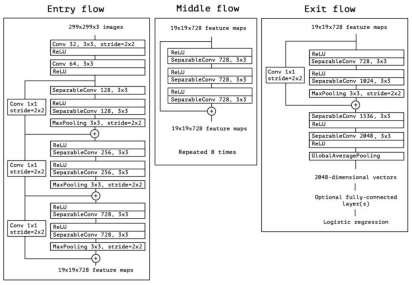


Figure 4. Xception architecture

3.1. Get image embedding

An image embedding is a lower-dimensional representation that can be extracted from a trained classification model. As the last layer is usually the fully-connected layer where the features are being classified, we can extract image features before the fully-connected layer of the model(as shown in Figure3). Figure 4 shows the entire Xception architecture. The image embedding is obtained by passing the image into the Xception model. The 2048-dimensional vector before the fully-connected layer at the output is the image embedding vector.

3.2. Neighbourhood embedding search

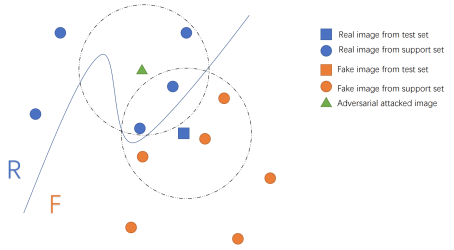
A good way for identifying similar images is to compute the cosine similarity between their image embeddings. Cosine similarity is computed between two vectors x_1 and x_2 in equation1.ϵ is a small positive number(1×10^{-8}) to avoid a potential division by zero.

$$\text{cosine similarity} = \frac{x_1 \cdot x_2}{\max\left(\|x_1\|_2 \cdot \|x_2\|_2, \epsilon\right)} \quad (1)$$

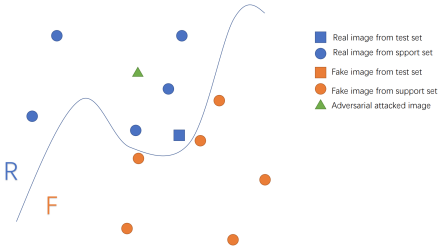
In this project, we implemented and tested on two image embedding search algorithms to find images in training set that are close to the test image: Neighbourhood adversarial embedding search and Neighbourhood embedding search for self-initiated adversarial attack sample.

3.2.1. Neighbourhood adversarial embedding search

Algorithm1 shows the details of Neighbourhood adversarial embedding search. The adversarial attacker will try to generate an input x_{adv} that is on the other side of the class boundary. The algorithm considers two anchor points: an input data x_0 and an adversarially attacked data x_{adv} near the decision boundary between class Real and Fake, as shown in Figure 5(a). Using Algorithm1, a batch of images that are near these anchor points are collected. The detection model is fine-tuned using this small batch of data before making a prediction. By using data points that are near the original and adversarial samples, the decision boundary of deepfake detection model will be refined as shown in Figure5(b).



(a) Neighbourhood adversarial embedding search



(b) Decision boundary after neighbourhood adversarial embedding search

Figure 5. Decision boundary before and after post-train using algorithm1.

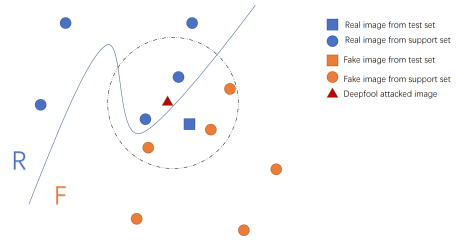
3.2.2. Neighbourhood embedding search for self-initiated adversarial attack sample

Algorithm2 shows the details of Neighbourhood embedding search for self-initiated adversarial attack sample. We use Deepfool attack on an input data x_0 to obtain its adversarial counterpart x_{df}. It is expected that x_{df} will be close to the decision boundary, as shown in Figure6(a). Through Algorithm2, a batch of images that are "near" the adversarial sample and decision boundary are collected and the detection model are trained using this small batch of data from training set before making a prediction. By using data points that are the adversarial sample and decision boundary, the decision boundary of deepfake detection model will be refined as shown in Figure6(b).

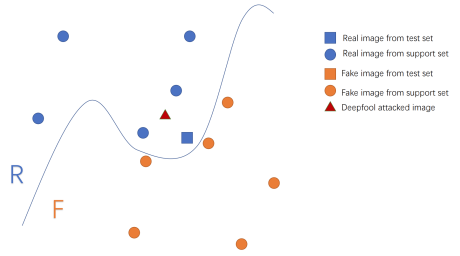
Algorithm 1 Neighbourhood adversarial embedding search

Input: trained detection model M, testing data x_0, adversarial attack image x_{adv} from pgd attack, training dataset T, similarity threshold τ
Output: neighbourhood embedding set N

Randomly sample a batch of data from training dataset T as support set S.
repeat
  Obtain model embedding of testing data x_0 as e_0
  Obtain model embedding of adversarial sample x_{adv} as e_{adv}
  for each image s_i in support set S **do**
    Obtain model embedding of image s_i as e_i
    Compute cosine similarity between e_i and e_0 as $Sim(e_i, e_0)$
    Compute cosine similarity between e_i and e_{adv} as $Sim(e_i, e_{adv})$
    if $Sim(e_i, e_0) > \tau$ or $Sim(e_i, e_{adv}) > \tau$ **then**
      Append corresponding image s_i from S to neighbourhood embedding set N.
    end if
  end for
until post-train iteration finished



(a) Neighbourhood embedding search for self-initiated adversarial attack sample



(b) Decision boundary after Neighbourhood embedding search for self-initiated adversarial attack sample

Figure 6. Decision boundary before and after post-train using algorithm2.

3.3. Fine-tuning the model during inference

After Neighbourhood embedding search, a batch of images is collected for post-training. The original classification model is fine-tuned using the small batch of data chosen using Algorithm1 and Algorithm2.

4. Experiments

4.1. Dataset

We tested our technique on a subset of FF++(Rossler et al., 2019). The FF++ dataset comes from 977 YouTube videos, all of which contain trackable and mostly unoccluded frontal face. We trained XceptionNet(Chollet, 2017) classification network as baseline model for evaluation.

Our results are shown in Table-1.

4.2. Experiments Setup for post-training

similarity thresholdτ:0.9, post-train optimizer:SGD with momentum:0.9 and learning rate:0.0001, post-train iteration:20, batch size:16.

5. Conclusion

In this project, we proposed a technique for Deepfake Detection to guard against adversarial attacks. Our technique is based on a post-training concept. Our experiments demonstrated that for a well-trained baseline deepfake detection model, adversarial attack will significantly drop its accuracy from 90.75% (denoted as Normal accuracy) to 1.04% (denoted as Robust accuracy). Using post-training will increase its performance in terms of robust accuracy with a little drop in Normal accuracy. We also experimented with freezing parameters before fully-connected layer during post-training. We found that when compared to partial-tuning (i.e. freezing parameters except the fully connected layers), the no-freezing approach (i.e. modifying all parameters in the model) always gave better results. Experiment results show that without retraining and introducing additional data, our technique can improve the robust accuracy for Deepfake detection models that are vulnerable to adversarial attacks.

References

Chollet, F. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. Deep-

Algorithm 2 Neighbourhood embedding search for self-initiated adversarial attack sample

Input: trained detection model M, adversarial attacked images from deepfool x_{df}, training dataset T, similarity threshold τ
Output: neighbourhood embedding set N

Randomly sample a batch of data from training dataset T as support set S.
repeat
  Obtain model embedding of adversarial attacked images from deepfool x_{df} as e_{df}
  for each image s_i in support set S **do**
    Obtain model embedding of image s_i as e_i
    Compute cosine similarity between e_i and e_{df} as $Sim(e_i, e_{df})$
    if $Sim(e_i, e_{df}) > \tau$ **then**
      Append corresponding image s_i to neighbourhood embedding set N.
    end if
  end for
until post-train iteration finished

fool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., and Nießner, M. Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1–11, 2019.

Method & Neighbourhood search method	freeze model embedding	Normal ACC	Robust ACC
Xception - Baseline	/	90.7567	1.0456
Xception - Algorithm 1	freeze	85.7914	4.4315
Xception - Algorithm 1	no freeze	85.2683	4.9954
Xception - Algorithm 2	freeze	84.6501	8.3898
Xception - Algorithm 2	no freeze	84.3871	9.4839

Table 1. The results are tested on subset of FF++(Rossler et al., 2019). Normal ACC is tested on test dataset without adversarial attack. Robust ACC is tested on test dataset with adversarial samples attacked by fgsm with ϵ=2/255. Xception is for trained Xception baseline model in 90.7567 Normal ACC. Algorithm 1 for Neighbourhood adversarial embedding search; Algorithm 2 for Neighbourhood embedding search for self-initiated adversarial attack sample. For "freeze model embedding" column, "freeze" means using post-train on model freezing all parameters EXCEPT fully connected layer;"no freeze" means using post-train on model without freezing parameters.

A. Appendix

Github: https://github.com/Yinjie-ZHENG/Adaptive-Modelling-for-Deepfake-Detection-using-post-train