Learning Face Anti-Spoofing with Feature Pretraining and Sequential Modeling

Pin-Ying Wu Pei-Ying Lin Zi-Ting Chou Chun-Tin Wu
National Taiwan University

Introduction

As more applications using face for user authentication are available, it is very important to develop algorithms for **face anti-spoofing**. The task of face anti-spoofing is usually defined as a **binary classification task**, where data sequences are attributed into two classes:

- **Real**: Face images which are directly captured by cameras.
- **Fake**: Face images which are remade from printed photos, replay-videos, etc.

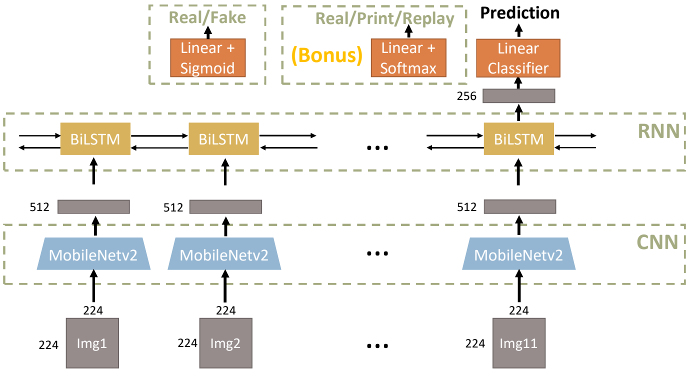
The goal is to predict whether input images are **Real** or **Fake**. As there are multiple ways to spoof the network, detecting these fake images is a challenging task. In addition to the binary prediction, for the **bonus** task, we also predict the type of the sequences, i.e. **Print, Replay and Fake**.

Highlights of this project:

- We adopt **weighted Focal Loss [2]** to address the data imbalance issue.
- We leverage **sequential modeling** to learn temporal information.
- We adopt **feature pretraining** to learn a more generalized feature.

Our model with all these components achieves **satisfactory performance on both Oulu-NPU (99.3%) and SiW (99.3%) dataset**. On the **bonus task, our method achieves 76.3% accuracy on SiW dataset.**

Method



Architecture: Our model is composed of two modules, a **CNN** to extract **spatial** features for input images and a **RNN** to learn **temporal** information.

1) **CNN:** We adopt **MobileNetv2** with ImageNet pretrained weights as the feature extractor, which is followed by a linear layer that converts the feature dimension from 1280 to 512.
2) **RNN:** We choose a two-layer **bidirectional LSTM** as our RNN module, since it can model sequential information bidirectionally. The final prediction is produced from passing the last output of RNN through a linear layer with **Sigmoid** and **Softmax** activation function for the **main task (Real/Fake)** and the **bonus task (Real/Print/Replay)** respectively.

Learning: Although the whole network can be trained end-to-end, the back propagation path for CNN could be very long, especially for the former layers of CNN. As a result, we divide our training process into **two stages**:

1) **Feature Pretraining**: To learn a **more generalized CNN feature**, we **augment** the data with the pipeline of ***random crop from 256x256 to 224x224***, ***random horizontal flip*** and ***random rotation within 15 degrees***. We adopt image-based sampling across all the sequences and treat each image as individual data for CNN input, which is trained for 30 epochs.
2) **Sequential Modeling**: In this stage, the **CNN and RNN** are trained **end-to-end** for 90 epochs, where the CNN is initialized with the pretrained weight from the first stage. We sample **8 sequences** per batch, each sequence consists of 11 image frames without using data augmentation. This stage aims to learn the **temporal information** of the data.

Loss: The quantity of **Real and Fake** training data is highly **imbalanced (1:4)**, which may lead to imbalanced classification problem. To address this issue, we adopt **weighted Focal Loss [2]** as our loss function:

$$FL(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t),$$

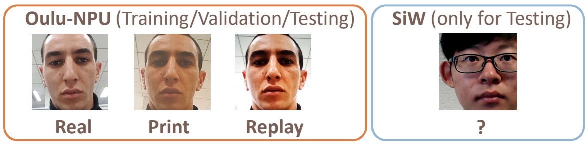
where p_t is the probability of the ground truth label, and γ controls the shape of the curve. The higher the value of γ , the lower the loss for well-classified examples. α_t is the weight for Real class, which is **0.8** in this case to balance the influence of the two classes, according to the ratio of Real and Fake data. For the bonus task, we use **0.5/0.25/0.25** as the weight for **Real/Print/Replay**.

Reference

[1] Y. Liu, A. Jourabloo, X. Liu. Learning Deep Models for Face Anti-Spoofing: Binary or Auxiliary Supervision. CVPR 2018.
[2] T. Lin, P. Goyal, R. Girshick, K. He, P. Dollar. Focal Loss for Dense Object Detection. ICCV 2017.

Dataset

Oulu-NPU: 11 frames are sampled from each video, which are captured by 6 phones, 3 acquisition conditions and 5 access types (Real, Print1, Print2, Replay1, Replay2). For our **main task**, the binary classification task, we label both **Print and Replay** as **Fake.**
SiW: 10 frames are sampled from each video.



Experiment

At first, we tried to reproduce a paper [1] which adopts RNN after CNN as its model. However, **they relied on the depth map and rPPG signals generated from other face pretrained models** to produce the prediction score, which is prohibited for this project. We then try another depth estimation model that is not trained with face data, but the result was not good. However, we learned a lot from this practice. We found that **the failed model simply guesses all the images to be Fake**, but still got an acceptable accuracy in training data due to the class imbalance.

To verify that our model is *trainable*, we did **data resampling to balance Real and Fake data** by using only ¼ of Fake data such that the quantities of the two classes are matched. We also simplified the model using a CNN with only 3 convolutional layers (Conv3) before the RNN, which reaches *77.0%* AUC on Oulu-NPU testing set and *65.6%* AUC on SiW testing set.

However, under this setting, we can only train with part of the training data. Instead, we searched for some reference to address the imbalanced issue without abandoning training data. We then adopted **weighted Focal Loss [2]** as our loss function, which highly improved the data imbalanced problem. The model achieved *81.4%* AUC on Oulu-NPU testing.

After deciding the loss, we experimented with **different architectures** for this task, including different architectures of CNN and RNN. For CNN, we finally chose **MobileNetv2** as our feature extractor, since it is known as more computationally efficient but still can reach a great performance. It achieves *98.9%* AUC on Oulu-NPU testing set, which passed the baseline. We then conducted experiment that replace the 1-directional LSTM with a **bidirectional LSTM (BiLSTM)**, and found that the model with the latter performs better (*99.4%* AUC on Oulu-NPU testing), since it can propagate the information bidirectionally.

Ablations on loss function (w/ MobileNetV2, 1-directional LSTM)

Loss	Val ACC	Val AUC	Oulu-NPU Test
BCELoss	95.1	99.8	98.7
weighted focal loss	94.4	100.0	98.9

Ablations on CNN (w/ weighted Focal Loss, 1-directional LSTM)

Architecture	Val ACC	Val AUC	Oulu-NPU Test
Conv3	93.0	97.5	81.4
Alexnet	92.8	96.8	82.1
ResNet18	96.4	99.9	96.4
ResNet34	98.3	100.0	97.5
MobileNetv2	94.4	100.0	98.9

Ablations on RNN (w/ weighted Focal Loss, MobileNetV2)

LSTM	Val ACC	Val AUC	Oulu-NPU Test
1-directional	94.4	100.0	98.9
bidirectional	98.6	99.9	99.4

When evaluating on SiW dataset, we encountered an issue that this model cannot generalize well to the unseen dataset, where we only reach *87.2%* AUC on SiW testing. We thought that **the back propagation path for CNN might be too long** so that the former layers of CNN are not learned properly. Therefore, as described in the method section, **we adopted feature pretraining with data augmentation in the first stage and train the whole network end-to-end in the second.** This successfully improved the performance on unseen dataset, which achieved ***99.3%* AUC on both Oulu-NPU and SiW testing set.** For the **bonus task**, we adopt the same method as the main task but change the final classifier to a Softmax classifier, which achieves *98.0%* accuracy on Oulu-NPU validation set and ***76.3%* accuracy on SiW testing set**.

Ablations on feature pretraining (w/ weighted Focal Loss, MobileNetV2, BiLSTM)

Feature Pretraining	Val ACC	Val AUC	Oulu-NPU Test	SiW Test
N	98.6	99.9	99.4	87.2
Y	97.9	99.9	99.3	99.3