Traccia 1: Valutazione della sicurezza di un sistema di face recognition

Obiettivo: Valutare la robustezza di un sistema di controllo accessi basato su face recognition ad adversarial attacks con intensità limitata e verificare la possibilità di attuare difese efficaci.

1. Scegliere 100 identità del training set del dataset VGG-Face2 (https://drive.google.com/file/d/1SXhc8m5PHxyM4lEWVEccSufIa8OiOjGW/view?usp=sharing) e costruire un test set di almeno 1000 immagini (almeno 10 per ogni identità), estraendole dal dataset (https://drive.google.com/file/d/1K56kVYHHDfLA2Anm7ga0tQolMwIPk6R8/view?usp=sharing) oppure ricercandole in rete.

2. Valutare l'accuratezza della rete di riconoscimento facciale fornita a lezione (da questo punto in poi NN1) sul test set così costituito

3. Generare degli adversarial examples utilizzando la libreria ART e NN1 come target. Valutare e discutere l'effetto di attacchi error-generic e error-specific su NN1, rivalutando l'accuratezza con opportune Security Evaluation Curves. N.B. Gli attacchi possono avere intensità massima Linf pari al 5% del range di valori rappresentabili (es. con valori tra 0 e 255 è circa 13, mentre con valori tra 0 e 1 è pari a 0.05).

4. Scegliere in rete un altro classificatore qualsiasi addestrato sul training set di VGG-Face2 (es. https://github.com/cydonia999/VGGFace2-pytorch) (da questo punto in poi NN2) e sperimentarlo sul test set "clean".

5. Verificare se gli adversarial examples hanno effetto anche su NN2 (trasferibilità).

6. Implementare e valutare le performance di almeno un meccanismo di difesa a scelta tra detector di adversarial samples e pre-processing.

7. Produrre un documento di dettaglio della sperimentazione effettuata, descrivendo le procedure di valutazione delle performance, la generazione degli attacchi, la generazione e la valutazione della difesa e commentando i risultati sperimentali. N.B. In fase di consegna, deve essere incluso tutto il codice e il/i dataset (es. condividendo una cartella su Google Drive)

Traccia 2: Realizzazione e validazione di un sistema di malware detection

Obiettivo: Realizzare e valutare la robustezza di un sistema di malware detection utilizzando metodologie di machine learning e deep learning.

1. Selezionare un metodo di machine learning e uno di deep learning tra i metodi visti durante il corso da impiegare come sistemi di malware detection.

2. Addestramento e validazione dei metodi selezionati.
   - Addestrare i classificatori scelti usando training e validation set del dataset 1
   - Valutare l'accuratezza dei metodi scelti sul test set del dataset 1
   - Valutare la capacità di generalizzazione dei metodi scelti sul dataset 2

3. Valutazione della robustezza dei metodi selezionati.
   - Generare dei campioni offuscati tramite l'attacco GAMMA per entrambi i metodi.
     - Per la selezione dei campioni da offuscare, si usino almeno 100 malware provenienti dal test set del dataset 1 che vengono correttamente rilevati come malware dal metodo da attaccare.
     - Si generino gli adversarial samples in diversi setup sperimentali in cui variano i parametri associati all'attacco.

4. Discutere di possibili metodi di difesa, senza necessariamente implementarli, da poter utilizzare per rendere robusti tali sistemi.

5. Produrre un documento di dettaglio della sperimentazione effettuata, descrivendo il setup sperimentale usato, le procedure di estrazione delle feature, di addestramento e validazione e giustificando i risultati sperimentali ottenuti negli step precedenti. In fase di consegna, deve essere incluso tutto il codice (es. condividendo una cartella su Google Drive)

Dataset:

- Dataset 1, da usare per la realizzazione dei sistemi di malware detection. I malware sono prelevati da SoReL-20M e disponibili in formato PE (non compressi). Ogni archivio contiene due cartelle contenenti rispettivamente i file malware e benigni di ogni set.
  - Training set: dataset1_train.tar.gz
  - Validation set: dataset1_validation.tar.gz
  - Test set: dataset1_test.tar.gz

- Dataset 2, da usare per valutare la capacità di generalizzazione dei metodi selezionati. Eventualmente, il contenuto benigno da iniettare nei malware con l'attacco GAMMA può essere prelevato dai benigni presenti in questo dataset.
  - VirusShareDataset.tar.gz

Note:

1. Usare la stessa versione di LIEF sia per l'estrazione delle features che durante la generazione dei campioni corrotti

2. Nella generazione dei feature vector di EMBER, scartare i feature vector dei file che LIEF non riesce ad analizzare correttamente (si catturi correttamente l'eccezione lanciata)