Anti-Facenet

Sam Margolis, Tushar Menon
sammargolis@wustl.edu, tusharmenon@wustl.edu

Abstract

FaceNet is the state of the art facial recognition system, developed by Google in 2015. Facial recognition is a highly weaponized concept that can, and has been used against public interest. We are thus motivated to formulate adversarial countermeasures against these systems through experimentation. In this work we implement FaceNet embedding for a dataset of celebrity faces and learn a classifier in the embedded space. We then attempt to attack it adversarially through poisoning the incoming data set. This could potentially

Introduction

Facial recognition technology is that which has the capability of identifying people from images or video. Advances in computer vision have led to systems that can achieve over 99.5% accuracy on the Labeled Faces in the Wild dataset and 95% on YouTube Faces DB. While technically impressive, these systems pose a serious threat to the security and privacy of the identifiable population.

This is seen in the recent development of the app 'Clearview AI', whose creator is said to have scraped millions of images from Twitter, Facebook, YouTube and other social media platforms to build a facial recognition system. Clearview AI is allegedly capable of identifying the layperson from data as bad as low quality degraded video from surveillance camera footage, overcoming the major facial recognition obstacles of lighting in-variance and pose invariance. The app is being sold to law enforcement departments across the world to identify suspects from surveillance footage. This is in itself is a problem, as is evidenced by events in Hong Kong, South America, India, etc. wherein law enforcement tries to move toward a police state and oppress their citizens. In that context, Clearview AI becomes a dangerously weaponized threat to freedom and democracy of the people. That notwithstanding, the creator of the app has also licensed the usage of his app to anyone willing to pay for it, which is dangerous for obvious reasons.

For these reasons, we are interested in formulating adversarial countermeasures to FaceNet that are not only effective but easy for a layperson to deploy, for instance, in the form of a filter applied to a photograph. In this area, there is considerable work on attacking Convolutional Neural Networks for facial recognition (Alparslan et al. 2020), as well as work in making deep networks robust to such attacks (Massoli et al. 2019). However, FaceNet is not a classifier neural network but it instead learns mappings of the faces onto embedded Euclidean space. A classifier can then be trained on these embedded representations and perform well on unseen data. Garafalo et al.2018, explore how to adversarially inject poisoned data into the dataset of such a classifier to hamper its accuracy. We aim to achieve a similar result, but not with the intention of just reducing the model accuracy, but to instead have the model be rendered useless through inaccurate but consistent classification. For instance, person A would always be identified as person B instead of being arbitrarily misclassified each time. We wish to identify the best strategy for poisoning the dataset in terms of 1. How many pictures of a given class label need to be poisoned? 2. What is the configuration of attacker class labels and attacked class labels that minimizes the amount of data poisoning required?

We are also interested in evaluating decision time attacks against the facial recognition systems, since a lot of the time these systems use facial recognition to identify faces on the internet and scrape them to append to their training data. Evasion attacks would then have the capability of not being picked up by these scrapers. It is also of interest to us to see what happens when these evasive instances do get picked up by scrapers and appended to the training data. In other words, what is the effect, on the classifier, of using a decision time adversarial instance as a poisoning instance instead?

We first implement a FaceNet classifier that can reliably classify celebrity faces. We are unable to gain access to the dataset that Clearview AI uses as it was acquired unethically. We instead choose to train our model on the 5 celebrity faces dataset hosted by Kaggle. This simplifies our problem as we can still investigate the answer to question 2 but the population of labels is vastly reduced. We used a FaceNet model to learn the mapping from images of faces to embeddings in Euclidean space. We then used a Multitask Cascaded Convolutional Network (Zhang et al. 2016) to perform face detection and isolate the region of the images that contained the faces. We then called this the training set and used a support

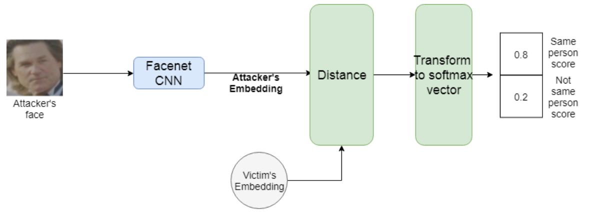

Figure 1: Facenet model with classification layer attached. by López et al. 2018

vector machine to learn a decision boundary in Euclidean space to perform classification on the celebrity faces. The results are as follows: The model has a 100% classification accuracy on the label-homogeneous testing set.

We then try to implement a few different methods of attacking this setup. The first, proposed by Biggio et al. 2013, is a poisoning attack on the support vector machine itself. The attack uses a kernelized gradient ascent strategy to maximize the loss of the SVM. The second method is an FGSM attack, proposed by Goodfellow et al., 2015. In the FGSM attack, we compute the gradient of the loss of the CNN according to the input image then add the sign of that gradient to the model weights. In the most basic sense, FGSM adds perturbations which the model believes to be important inputs.

Our contribution is thus:

- We explore adversarial attacks in the domain of facial recognition, where there has not been much former work in this area.
- We investigate experimentally whether decision time attack instances can be used to poison a dataset, and if so, how?

Model

The target model is a pretrained Keras FaceNet model by Hiroki Taniai, 2018. We use this to learn the mappings from the image space to Euclidean space of the Kaggle 5 celebrity faces dataset. These embeddings turn the abstract representations into feature vectors that numerically capture the facial characteristics of the input images. These numerical forms can later be augmented into a training dataset for a classifier. Here is the outline of the FaceNet workflow:

1. Face detection - We first employ a Multi-task Cascaded Neural Network to detect the presence of a face in the image (Zhang et al. 2016)
2. Preprocessing - We apply an affine transformation to the image to convert it to a format convenient for the neural network. The eyes, nose, and other facial features are preserved at the same location on every image
3. Feature extraction using a convolutional neural network - The CNN iterates through samples of the same person and calculates the Euclidean similarity between features of each sample (for a given set of features). It then has a notion of what those features should look like for that person and therefore a different person would have feature values that are more distant from person A in the l2 sense.

The template of set of features used by the convolutional neural network is developed by Schroff et al. in the FaceNet paper.

The support vector machine is then learned by minimizing the hinge loss on a training set $\mathcal{D}_{tr} = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$:

$$\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \sum_{i=1}^{n} \left(1 - y_i \mathbf{w}^T \mathbf{x}_i\right)_+ = \sum_{i=1}^{n} (-g_i)_+$$

To execute a poisoning attack, the adversary's problem is then to curate a validation set $\mathcal{D}_{val} = \{\mathbf{x}_i, y_i\}_{i=1}^{m}$ and craft a point $\mathbf{x}_c$ that maximizes the above loss on the validation set:

$$\max_{\mathbf{x_c}} \mathcal{L}(\mathbf{x_c}) = \sum_{k=1}^{m} \left(1 - y_k \mathbf{w}^T \mathbf{x}_k\right)_+ = \sum_{k=1}^{m} (-g_k)_+$$

(Biggio et al. 2013) showed that the above problem of computing the gradient of $\mathcal{L}(\mathbf{x_c})$ can be solved by exploiting properties of SVMs. In short, the gradient is calculated with respect to the direction of the attack vector u as:

$$\frac{\partial L}{\partial u} = \sum_{k=1}^{m} \left\{ M_k \frac{\partial Q_{sc}}{\partial u} + \frac{\partial Q_{kc}}{\partial u} \right\} \alpha_c$$

where Q is $\mathbf{y}\mathbf{y}^T K$ where K is the kernel matrix. M_k is defined as:

$$M_k = -\frac{1}{\zeta} \left(Q_{ks} \left(\zeta Q_{ss}^{-1} - vv^T \right) + y_k v^T \right)$$

where $v = Q_{ss}^{-1} y_s$ and $\zeta = y_s^\top Q_{ss}^{-1} y_s$. The poisoning attack is then described by the algorithm in the following section.

In targeted FGSM, we are trying to add noise to the input such to drift the inputs classification away from the true label and towards another. If $\boldsymbol{\theta}$ are the model parameters, and $J(\boldsymbol{\theta}, \mathbf{x}, y)$ is the cost function, then Goodfellow et al. 2015 show that we can linearize the cost function about the current values of $\boldsymbol{\theta}$, obtaining the following perturbation η.

$$\boldsymbol{\eta} = \epsilon \operatorname{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right)$$

We can iterate on the following to drift the image:

$$\mathbf{x}' = \mathbf{x} - \epsilon \operatorname{sign} \cdot \left(\nabla_x J\left(\mathbf{x}, \mathbf{x}_t\right)\right)$$

To determine whether the model is fooled and predicts either that it is or isn't a certain class, we can use soft max which will return a number between 0 and 1 on whether it believes the face.

$$\hat{y} = \operatorname{softmax}(\boldsymbol{\theta}^\top \mathbf{x})$$

Algorithms

The high level idea behind the poisoning attack on the SVM is as follows - The attack vector x_c is picked arbitrarily from the set of points in the attacked class' margin and its label is flipped. x_c is then optimized iteratively through Algorithm 1. A drawback of this attack is that it requires the attacker to be perfectly knowledgeable about the model. That is to say, the attacker must know the feature extraction done by the CNN, the hyperparameters of the SVM and the SVM weights, as well as access to the training data. Furthermore the attacker is also presumed to have the ability to modify the training data and inject sample into it.

```
Algorithm 1: Poisoning attack on SVM
  Input:$\mathcal{D}_{tr}, \mathcal{D}_{val}$
  Output:$x_c$, the attack vector
  1. Initiliaze attack vector from attacked class margin
     and flip label to $y_c$ ;
  2. $\{\alpha_i, b\} \leftarrow$ Learn an SVM on $\mathcal{D}_{tr}$
  3. $k \leftarrow 0$
  while $L\left(x_c^{(p)}\right) - L\left(x_c^{(p-1)}\right) < \epsilon$ do
      4. Find an SVM solution on $\mathcal{D}_{tr} \bigcup \{x_c, y_c\}$ using
         Algorithm 2;
      5. Calculate $\frac{\partial L}{\partial u}$ on $\mathcal{D}_{\text{val}}$ and set $u$ to be a unit
         vector aligned with the gradient;
      6. $k \leftarrow k + 1$ and $x_c^{(p)} \leftarrow x_c^{(p-1)} + tu$
  end
  7. return $x_c$
```

The $\{\alpha_i, b\}$ in step 2 refer to the dual coefficients of the SVM optimization.

Algorithm 2 is a method of recursively training support vector machines, developed by (Cauwenberghs and Poggio, 2001). From the first order conditions on the dual formulation objective of the SVM optimization, we get the Karush Kuhn Tucker conditions:

$$g_i = \frac{\partial W}{\partial \alpha_i} = \sum_j Q_{ij}\alpha_j + y_i b - 1$$

$$= y_i f(\mathbf{x}_i) - 1 \quad \begin{cases} \geq 0; & \alpha_i = 0 \\ \leq 0; & 0 < \alpha_i < C \\ \leq 0; & \alpha_i = C \end{cases}$$

This partitions $\mathcal{D}_{tr}$ into three categories: The set of margin support vectors S, the error support vectors E and the set of reserve support vectors R. Taking $\mathcal{R}$ to be the current Jacobian inverse of Q, Algorithm 2 proceeds as follows:

```
Algorithm 2: Incremental Learning
  Input:$\mathcal{D}_{tr}, x_c, y_c, \{\alpha_i, b\}$, S, R, E
  Output:Updated S, R, E
  1. Initiliaze $\alpha_c$ to 0 ;
  2. if $g_c > 0$, terminate ($x$ is not a margin or error
     vector) ;
  3. if $g_c \leq 0$, increase $\alpha_c$ such that one of the
     following occurs:
  (a) $g_c$ = 0: Add $c$ to margin set S, update $\mathcal{R}$ accordingly, and
      terminate ;
  (b) $\alpha_c = C$: Add c to error set E , and terminate ;
  (c) Elements of $\mathcal{D}_{tr}$ migrate across S, E, and R: Update
      memberships and $\mathcal{R}$ accordingly ;
```

Using the two algorithms as described above should yield a point x_c capable of adversarially poisoning the FaceNet classifier.

The method of attacking using FGSM is entirely different. The first broad difference is that it is an evasion attack rather than a poisoning attack, even though we will use the adversarially crafted instances as poisoning instances. Secondly, in the context of our problem, the FGSM attack is directed toward the embeddings learned by the deep neural network FaceNet themselves. This is in contrast to the attack described above, which does not touch the FaceNet embeddings and instead tries to attack the support vector machine classifier that uses the embeddings, via poisoning.

In short, the fast gradient sign method operates

$$x' = x - \epsilon \,\text{sign} \cdot \left(\nabla_x L\left(x, x_t\right)\right)$$

The end result becomes a half way point between mislabeled data and the clean label since the input image only drifts towards another individual in the set.

```
Algorithm 3: FGSM
  Input:$x, \epsilon, x_t$
  Output:$x'$
  1. Initiliaze $x'$ to $x$ ;
  2. Take the gradient of the loss function between $x$
     and $x_t$;
                    $\nabla_x L\left(x, x_t\right)$
  3. Determine the sign of the gradient for each tensor
  4. Multiply that sign by the predetermined $\epsilon$ and
     subtract it from $x$
  5. Repeat this in iterative manner until you are
     satisfied with the outcome
```

Results/Evaluation

We first evaluate the untampered FaceNet classifier (the support vector machine). (Schroff et al. 2015) first proposed the idea of FaceNet, and their implementation achieved over 99.5% accuracy on the Labeled Faces in the Wild dataset. As mentioned before, for simplicity's sake, we are restricting our data to contain only 5 categories of class labels as opposed to LFW's $\sim$ 1300. So we anticipate our implementation to perform at least as well, if not better. And indeed, our implementation of the FaceNet classifier achieves 100% classification score on Kaggle's 5 celebrity faces dataset. This is still a harsh metric because in multi-class classification, it requires that for each sample each label set be correctly predicted.

For the poisoning attack on SVM, we evaluate the how adding a single poisoned instance to the training data affects the overall model score. We compute how the overall model score changes as the strength of the SVM changes. This is done in two layered way. First, we vary the hard margin tolerance of the SVM (v). We then interface this with varying the size of the training set of the SVM. Naturally, we expect weaker SVMs to be more susceptible. As tertiary explorations, we are also interested in how the overall model score changes as the number of poisoned instances for a given class label increases. Lastly, we wish to evaluate what number of instances in what number of class labels

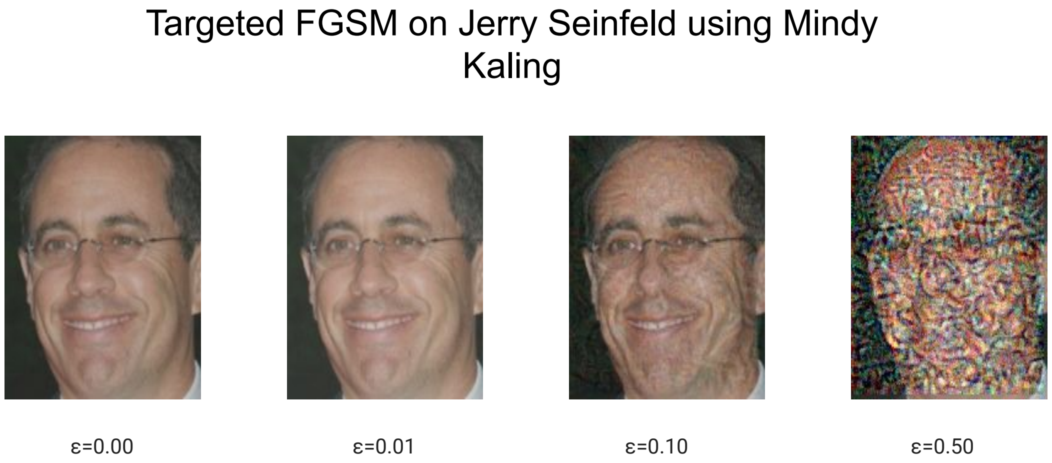

Figure 2: Images of Jerry Seinfeld under an FGSM attack. This was set up using cleverhans and a picture of Mindy Kaling.

poisoned class labels yields the SVM classifier unreliable (performance similar to random guessing). The table below displays our empirical results as we vary the hard margin tolerance v of the trained SVM as well as the size of the training set. We then attack each of these SVMs using Biggio et al. algorithm for SVM poisoning. Bear in mind each of these SVM's had perfect classification accuracy on the (relatively small) dataset of celebrity faces before the attack. We obtained the following results:

v	size	initial error	error after attack
0.05	5	0.0	1.32
0.05	10	0.0	1.12
0.05	15	0.0	0.79
0.05	20	0.0	0.34
0.1	5	0.0	1.42
0.1	10	0.0	1.88
0.1	15	0.0	0.32
0.1	20	0.0	0.19
0.15	5	0.0	1.15
0.15	10	0.0	0.91
0.15	15	0.0	0.48
0.15	20	0.0	0.12
0.2	5	0.0	1.21
0.2	10	0.0	0.88
0.2	15	0.0	0.34
0.2	20	0.0	0.09

It is evident that the SVM poisoning attack is succesful, as the error is raised from 0 to around 1 misclassification on average, which is about 5% of the dataset. FaceNet's accuracy on untampered data is 99.7% in Schroff et al. and our implementation scored 100% on the celebrity faces dataset, so dropping to around 95% is nontrivial and is a significant step forward. However, the data clearly indicate that as the strength of the SVM increases, it becomes harder to attack. This is seen in Figure 3.

We did implement the FGSM attack on the data set by building in an additional classification layer to the Facenet model. To do this, we began with David Sandberg Facenet implementation and the cleverhans library. Although the most traditional approach is to use the LFW data set which we did first train on, we then transferred to use the 5 celebrity data set from before to aid in efforts to poison. Since we wanted to determine the accuracy in a system that used facenet as embedding and a SVM as a classifier, we transfered the images which had FGSM done on them over to the model which used the Facenet for embeddings and an SVM for classification. This also helped to test how transferable the attack was.

There were a few challenges we believe we ran into here. First and foremost being that the attack isn't robust against the SVM because when we conducted FGSM, it was against a fully connected Keras NN but minimizing for the SVM carries a different problem. The next challenge we faced was that the embeddings may have helped to normalize out some of gradient steps taken.

There was no overall loss of accuracy when using a single poisoned example against the model which was to be expected. The more surprising result was that there was little to no decrease in the probability of a certain class even when attacking that class directly with the high ϵ values. Summarized below are the results of the average validation images predict probability results when the training data set is poisoned with different levels of ϵ

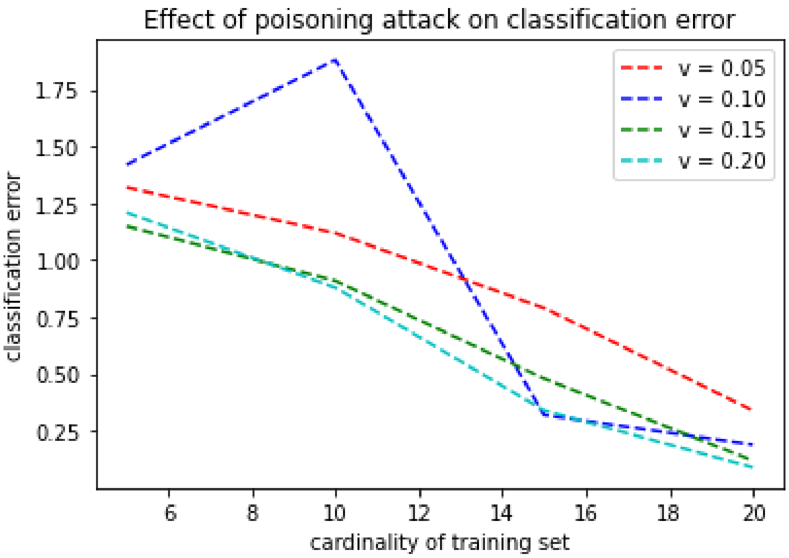

Figure 3: The average classification error for varying hyperparameters v and size of training set. The classification error is evaluated before the attack and after the injection of the poisoned data

ϵ	Avg Predict Probability for Jerry Seinfeld
0.00	86.4
0.01	86.3
0.10	85.7
0.50	84.5

The key reason we believe that it has not been effective thus far is that because only a portion of the data was poisoned, it just added more variance to the data set and made the classifier stronger. This is somewhat equivalent to iterative retraining. This was a serious flaw in our theory and a primary reason for its lack of success. We were not truly poisoning the classifier as it already had data which it was able to use to classify the faces effectively. What we were doing was simply adding additional data to the set which then allowed the classifier to handle FGSM attacks more effectively.

Related work

While there has not been copious amounts of work covering adversarial attacks on facial recognition from the angle that we approach it, it has indeed gathered the attention of adversarial machine learning researchers. These efforts have mainly taken the form of evasion attacks against facial recognition software. Sharif et al. (2016) developed a white box evasion attack that enable an attacker to not only evade detection but also impersonate other people. They also develop evasion attacks using generative adversarial networks to sample from a latent distribution of facial features.

The first major work we rely on is *FaceNet: A Unified Embedding for Face Recognition and Clustering* by Schroff, Kalenichenko and Philbin, 2015. This work is a huge milestone in the field of facial recognition. It proposes a system that directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. This is highly useful as now classifiers can be trained on these FaceNet embeddings of facial images instead of the space of facial images. These embeddings are powerful and exhibit the highly useful properties of pose invariance and lighting invariance, two of the major obstacles in the field. We use this model to learn embeddings for the 5 celebrity faces dataset hosted by Kaggle, and then to train a support vector machine classifier on these embeddings.

We also make heavy use of the poisoning algorithm proposed by Biggio et al. 2013 for attacking an SVM. They evaluate their attack on the MNIST dataset as opposed to facial images. They were able to demonstrate that the classification error rose from less than 1% to around 30% for both the testing and validation sets.

Garofalo et al. 2018 perform a similar experiment to us by trying to poison a SVM trained on facial images. They achieve significant degradation in AUC, to the point where the classification accuracy of the facial recognition system reached 50%, making it useless. This work differs from ours in that their goal is to simply reduce the classification accuracy whereas ours is to specifically get the support vector classifier to consistently mislabel instances from a given class as the attack class. Furthermore, our work explores a variety of other avenues for attacking the FaceNet model as opposed to just the one poisoning attack on the SVM.

The equalAIs lab at MIT pursued a similarly motivated project - they wanted to break facial detectors in the interest of privacy and security. Their work however, only focuses on how to get facial detectors to not detect faces, i.e, they are not trying to get a classifier to mislabel person A as person B, they are trying to get the detector (the Multi Task Cascaded neural network in our case) to not pick up on the existence of a face in an image. Secondly, their attack is a

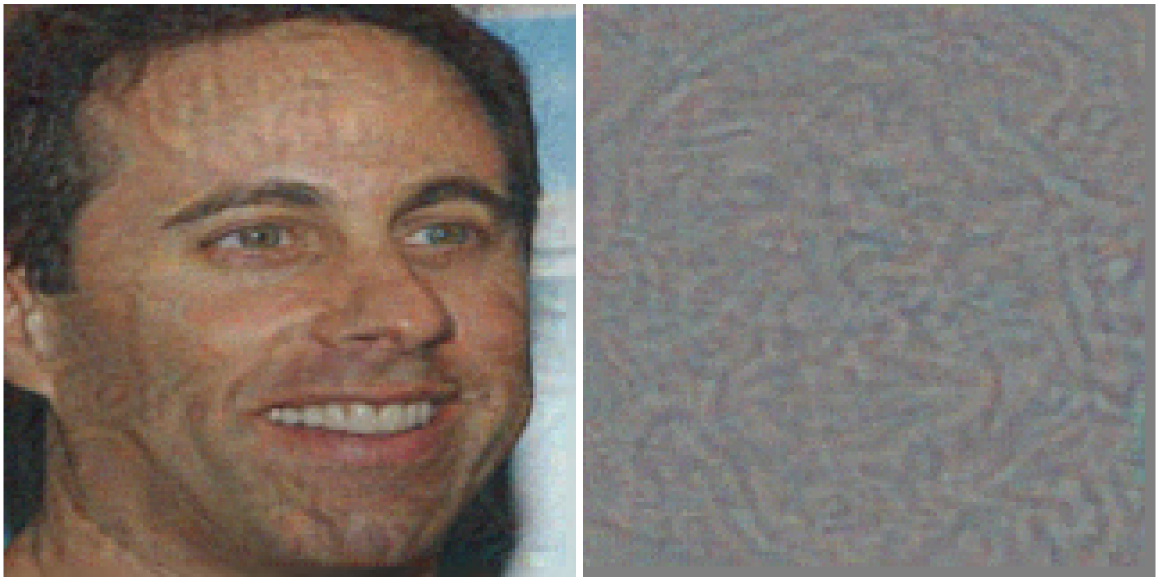

Figure 4: The adversarial image and noise

decision time attack, and their adversarial instances have a very perceptible equalAIs watermark on them.

For the FGSM attack, we obviously relied on the problem formulation specified by Goodfellow et al. 2015. The paper illustrates how deep networks can exploited and adversarial decision time examples can be created using a variety of techniques, FGSM being one of them. There is not much work that we were able to find that explored the effect of adding FGSM perturbed images to a training set as a poisoning attack.

Conclusions, Limitations, Future work

In summary, we have so far found that the FaceNet embeddings (and a SVM trained on them thereof) are not susceptible to using FGSM to poison the classifier. The reason for this could be that the FaceNet embeddings are implicitly robust in their high level feature abstraction. As demonstrated by Schroff et al.2015, the predictions of FaceNet do not change with significant changes made to a person's face image (different lighting, poses, backgrounds, etc.) This shows that the embeddings capture information about the facial features that are not weak to adversarial imperceptible noise, i.e, those feature relationships are preserved. A significant limitation for us here was the fact that we used a pretrained Keras FaceNet model by Hiroki Taniai, 2018, which was essentially a black box to us. This made it harder for us to attack.

Furthermore, in the process of deploying the FGSM attack, we were able to ascertain that decision time attack instances are not effective when employed to poison a dataset at training time. The reason for this could be that decision time attacks rely on exploiting the decision boundary of the classifier whereas when they are used to poison they merely strengthen the classifier (because they are causing it to train the decision boundary that would have been exploited earlier).

In executing the SVM poisoning attack, we found that the classifier is indeed vulnerable to an SVM attack. The success of the attack however relies on the strength of the SVM. These attacks are computationally expensive to generate and even more so when the SVM is trained on a rich dataset with rigid margin classification. Furthermore, our work only generates an adversarial feature vector in the template of the FaceNet CNN to fool the SVM classifier. We do not produce an image nor do we produce a filter that could be applied to an image. This is a route for potential future work - formulating a reverse mapping alike to Schroff et al. to go from the feature space to the original input space.

We many more questions around subsets of images which then render a model incompatible. It would be interesting to consider how we may strengthen the SVM attack to the extent that is able to violate the integrity of the system in total, which is to say the facial recognition system performs no better than random guessing.

One idea that we do have is to poison every one of the images in the data set with the same FGSM screen which hopefully would then be similar to the individual doing it at run time. Again to our surprise, there was very little reduction in accuracy. This surprised us but what we determined is that it may be because the gradient decent steps were not the same each time and thus again it added general variance rather than noise in a specific direction. If we did this again we would layer the exact same noise on each of the training photos so that the average would not be the same with more variance but rather the classifier would be altered.

This approach of adding the exact same noise on top of each image that we were adding in likely would have been much more effective because the classifier would have then learned the mapping of the poisoning. The problem with both of our approaches is that the noise was not learned since it was different each time and the other aspects of the image were more consistent one to another. With more time this is the approach we would have taken.

References

Schroff, F. ; Kalenichenko, D.; Philbin, J. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering in *Conference on Computer Vision and Pattern Recognition 2015*

Alparslan, Y.; Keim-Shenk, J.; Kade, S.; Greenstadt, R.; Adversarial Attacks on Convolutional Neural Networks in Facial Recognition Domain

Massoli, F.; Carrara, F.; Amato, G.; Falchi, F.;Detection of Face Recognition Adversarial Attacks, pre print

Garofalo, G.; Rimmer, V.; van Hamme, T.; Fishy Faces: Crafting Adversarial Images to Poison Face Authentication in *Usenix 2018*

Zhang, K.; Zhang, Z.; Li, Z.; Qiao, L.; Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks in *IEEE Signal Processing Letters 2016*

Sharif, M.; Bhagavatula, S., Bauer, L., Reiter, M. K.; Adversarial Generative Nets: Neural Network Attacks on State of the Art Face Recognition

Biggio, B.; Nelson B; Laskov, P; Poisoning Attacks against Support Vector Machines

Goodfellow, I.; Shlens, J.; Szegedy, C.; Explaining and Harnessing Adversarial Examples

Zhang, K.; Zhang, Z.; Li, Z., Qiao, Y.; Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks

Sharif, M.; Bhagavatula, S., Bauer, L., Reiter, M. K.; Accesorize to a crime: Real and Stealthy Attacks on State of the Art Facial Recognition