GlanceAgain: Adversarial Defense using Recurrent Layers

Anudeep Das
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
a38das@uwaterloo.ca

***Abstract*—Adversarial attacks against neural networks are a significant concern for ML researchers and designers, and defending against them continues to be a challenge. This paper presents a novel approach, "GlanceAgain", for defending against adversarial attacks targeting ML computer vision models. In GlanceAgain, the initial input image is incrementally transformed, passed through the original vision model, and then input through two Long Short-Term Memory (LSTM) layers, finally resulting in a prediction. In order to test the approach, adversarial samples were generated (using the Fast-Gradient-Sign-Method (FGSM) attack) from the original target model, and the accuracy of GlanceAgain on these samples was analyzed. The performance of GlanceAgain was also compared to adversarial training. All testing was conducted with the MNIST handwritten digit dataset. It was found that GlanceAgain performed much better (91%) on the adversarial samples than the original model (6%). In fact, GlanceAgain's performance was close to that of the adversarially trained original model (94%). GlanceAgain and the adversarially trained original mdoel were the defense models in this case. Further testing was also done to analyze the defense models' performance against adversarial samples crafted to specifically attack the defense models themselves. And unfortunately, the accuracy of GlanceAgain against its own crafted samples was lower than the accuracy of adversarial training against its respective crafted samples (51% compared to 87%). However, the performance was still significantly higher than the 6% achieved by the original model.**

***Index Terms*—adversarial attack, adversarial defense, security, vision models, LSTM**

I. Introduction

From online advertising, to facial recognition algorithms in surveillance, machine learning (ML) is becoming more ubiquitous around the world. As a result, there is growing concern about adversarial attacks against ML algorithms. An area where adversarial attacks have the potential to be especially devastating is in ML computer vision models. Notably, researchers at MIT were successfully able to conduct an attack against the vision model used in a self-driving car, and cause it to mistakenly interpret a stop sign as a speed limit sign [1]. The repercussions of this are highly dangerous since the self-driving car would fail to stop at the stop sign if it interprets it as a speed limit sign. As such, siginificant research has been done to explore defenses against adversarial attacks for vision models in particular. However, many of these methods only consider a single image as input to a vision model [3] [5]. This is a limitation since, as in the case of the self-driving car, many computer vision models have to process streams of images. Furthermore, the images in the streams have a logical sequence to them. For example, as a self-driving car approaches a stop sign, the stop sign appears to become progressively larger in the images of the car's input image stream. These logical sequences could even be generated by iteratively transforming a single input image. Furthermore, analyzing the semantics of sequences is a primary use-case of recurrent neural networks (RNN's) [2]. Additionally, and more importantly, it has been shown that the adversarial perturbations generated in an adversarial attack are sensitive to transformations [3]. Hence, in GlanceAgain, the model produces a sequence of images where each image is a rotation or scaling of the previous image in the sequence, starting with the initial input. Then, this sequence is passed through the vision model. For this paper, it is assumed that the vision model is comprised of only a fully-connected neural network, not a Convolutional Neural Network (CNN). The system also does not re-train the entire vision model, hence reducing the training time of GlanceAgain significantly. After that, the output of the vision model is passed through two Long Short-Term Memory (LSTM) units, and finally outputs the prediction. The specific vision task analyzed in this paper was the classification on MNIST handwritten digits, and the attack that was used was the Fast-Gradient-Sign-Method (FGSM) attack. It was found that GlanceAgain outperformed the original model on adversarial samples generated from the original model (91% compared to 6%). GlanceAgain's performance was close to that of the adversarially trained original model (94%). GlanceAgain and the adversarially trained original mdoel were the defense models in this case. Further testing was also done to analyze the defense models' performance against adversarial samples crafted to specifically attack the defense models themselves. And unfortunately, the accuracy of GlanceAgain against its own crafted samples was lower than the accuracy of adversarial training against its respective crafted samples (51% compared to 87%). However, the performance was still significantly higher than the 6% achieved by the original model. The contributions of this work are (i) a method to convert a single image prediction task into a multiple image prediction task via transformations

of the initial image, (ii) a study of adversarial attacks against models that iteratively transform their input images and pass them through a recurrent component.

II. Related Work

A. Adversarial Attacks

There are several different adversarial attacks that can be conducted against neural networks (referred to as "target networks"). This paper focused on evaluating GlanceAgain against the Fast-Gradient-Sign-Method (FGSM) attack [4].

- *Fast-Gradient-Sign-Method (FGSM) Attack*

The FGSM attack assumes that the target network was trained using gradient descent, and it leverages this assumption to perturb input samples in order to attack the target network, F_t. An adversarial sample $\tilde{x}$ can be constructed from an input sample x as

$$\tilde{x} = x + \epsilon \text{sign}(\nabla_x L(\theta, x, y))$$

where θ represents the parameters of the target network, $\text{sign}(\cdot)$ is the sign function, y represents the ground truth label, and $L(\theta, x, y)$ is the loss of the target network with respect to these values. ϵ controls the magnitude of the perturbation added to the input. To be more precise, the adversarial sample $\tilde{x}$ is such that

$$||\tilde{x} - x||_\infty \leq \epsilon$$

Essentially, the attack uses gradient ascent to perturb x so the target model misclassifies it. This variant of the FGSM attack is "untargeted". This is because the adversary does not care about the exact value of the target model's prediction on $\tilde{x}$, $F_t(\tilde{x})$; they only require that a misclassification occurs (ie. it should be the case that $F_t(\tilde{x}) \neq y$).

The targeted variant uses the following equation

$$\tilde{x} = x - \epsilon \text{sign}(\nabla_x L(\theta, x, y'))$$

In this case, $y' \neq y$ is selected as the target, and the adversary would like $F_t(\tilde{x}) = y'$.

A visual representation of this attack is shown in Fig. 1.

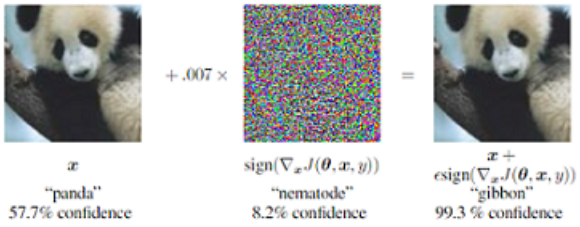

Fig. 1. An example of an adversarial sample generated from the image of a panda.

In Fig. 1, $\epsilon = 0.007$ is used. Despite looking almost visually identical to the human eye, the perturbed panda is misclassified as a gibbon with 99.3 % confidence by the target model. This is often the objective of adversarial attacks against vision models; the adversary attempts to generate samples that, from a human perspective, look nearly identical to the unperturbed samples, but they are misclassified by the target model.

This paper focuses on the untargeted FGSM attack.

B. Adversarial Defenses

This subsection discusses other adversarial defences that inspired GlanceAgain, or were used as comparisons for this paper.

- *Defense in [3]*

Many different defenses have been proposed to counter adversarial attacks. The work that inspires this paper the most is [3]. In their work, inputs to a CNN were randomly padded and resized prior to feeding them through the network. Their formulation is shown in Fig 2.

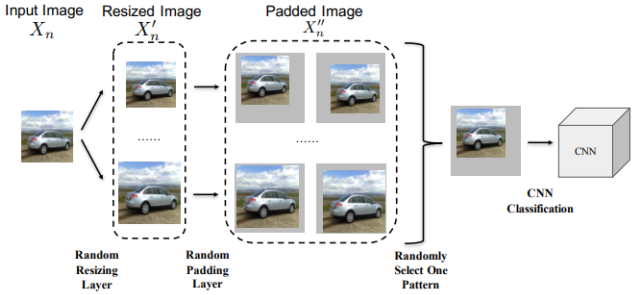

Fig. 2. The method used by [3] to generate randomly transformed images as an adversarial defense [3, Fig. 2].

Their theory was that transformations to the images "may... destroy the specific structure of adversarial perturbations" [3, p. 2]. Our paper extends this core idea to rotation as rotation was not investigated in their paper. Our paper further adds a recurrent component to the system because passing through several layers of transformations may destroy the adversarial perturbations even more.

- *Adversarial Training*

Another state-of-the-art adversarial defense method is adversarial training. This method entails a min-max optimization where images are first maximally perturbed, and then gradient descent optimization is conducted on this perturbed image [5]. The following is the mathematical formulation for adversarial training. The objective is to optimize $\rho(\theta)$, the adversarial loss, by finding

$$\min_\theta \rho(\theta) \text{ where}$$

$$\rho(\theta) = \mathbb{E}_{(x,y)\sim D}[\max_\delta L(\theta, \delta, y)],\ \delta = ||\tilde{x} - x||_\infty \leq \epsilon$$

where $\tilde{x}$ and $L(\cdot)$ are defined as previously. The expectation is taken with regards to samples from the training distribution D.

Another notable defense method is High-Level Guided Denoiser (HGD) [6]. In this method, a neural network learns to denoise the perturbations in an adversarial sample. That is, given an adversarial sample x^*, HGD learns $d\hat{x}$, the noise term, and generates a reproduction of the unperturbed image $\hat{x}$.

$$\hat{x} = x^* + d\hat{x}$$

A major strength of GlanceAgain is that other advanced and effective methods such as HGD could be incorporated into our model as another type of transformation. For example, to incorporate HGD, HGD's de-noising ability could be used as a transformation. This is an idea for future work.

C. Long-term Recurrent Convolutional Networks (LRCN's)

The structure of GlanceAgain was based largely on the structure of Long-Term Recurrent Convolutional Networks (LCRN's) [2]. In this architecture, sequences of video frames were passed through one or more CNN's before passing through an LSTM. The difference in our model is that GlanceAgain generates the images as a result of transformation, and they are not extracted from a video, and GlanceAgain currently does not use a CNN. Also, the task of GlanceAgain is simpler since it does not need to output a natural language sentence. A configuration for an LRCN is shown in Fig 3.

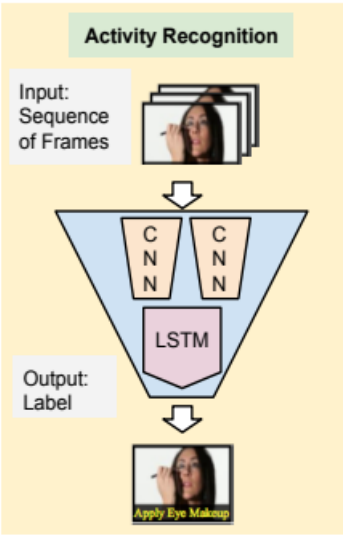

Fig. 3. The LRCN method applied to a sequence of frames [2, Fig. 3]

III. Proposed Method

The proposed method has three major components; image batch generation, feed-forward through a fully-connected network (the original model), and LSTM prediction.

A. Image Batch Generation

To generate an image batch the following transformations were used. The system does three rotations, then scaling, and then a final rotation, in this order. An image was obtained after each transformation (rotation or scaling). The output image from the previous transformation was used as the input image for the next transformation. In the end, we obtain a batch of 6 images, including the initial image. All of these were done with torchvision transformations.

B. Feed-Forward through a Fully-Connected Network

It is assumed that the original computer vision model is a fully-connected neural network. A fully-connected neural network consists of multiple layers where the output of one layer transforms the output of the previous layer starting from input data SOURCE. Specifically, the output of layer l with input x_{l-1} is

$$f(W_l x_{l-1} + b_l)$$

where W_l and b_l the weights and bias of the layer l, in matrix form f is a non-linear activation function such as sigmoid. Each layer consists of a number of neurons, whose weights and biases are all incorporated in W_l and b_l respectively. GlanceAgain feeds the batch of transformed images through the original vision model, however, the last layer of the model had to be changed. This is because the original vision model predicted a label in it's output layer, hence it used softmax activation and a relatively small number of nodes. However, in GlanceAgain, the vision portion of the model should output a hidden state with more nodes so as to conform with the input size of the subsequent LSTM layer. Furthermore, since this vision model is a fully-connected neural network, the 2-dimensional transformed images in the batch had to be flattened before being passed through this model.

C. LSTM Prediction

The LSTM is a type of recurrent neural network. Its design allows a neural network to have "memory" of the inputs in a sequence. Compared to a standard RNN, the LSTM is less susceptible to the phenomenon of vanishing gradient, and hence acheives better performance [2]. The reason for this is because there are logical gates within an LSTM that controls how much of an input sequence is used to generate a prediction. It is the use of long input sequences in RNN's that results in vanishing gradient [2]. The formulation of an LSTM is shown in Fig 4.

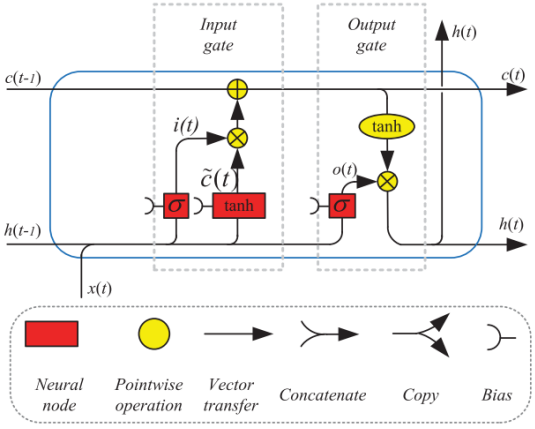

Fig. 4. The basic schematic of an LSTM cell [8, Fig. 2]

From Fig. 4, several algorithms are evident. The following explanations are paraphrased from [9]. Here, x_t is the input at time t (ie. it is the t-th input in the input sequence). Additionally, h_t is the hidden state of the cell at time t. More importantly, an LSTM has a context c_i which controls how much of a sequence the system incorporates into its memory. There are 3 main gates to an LSTM unit, which are defined as

$$f_t = \sigma([x_t h_{t-1}]W_f + b_f)$$

which is the forget gate,

$$i_t = \sigma([x_t h_{t-1}]W_i + b_i)$$

which is the input gate, which controls whether the input at time t gets incorporated into the context, where the input is defined as

$$\tilde{c_t} = \tanh([x_t h_{t-1}]W_c + b_c)$$

and the output gate

$$o_t = \sigma([x_t h_{t-1}]W_o + b_o)$$

which controls the output. Finally, we get

$$h_t = o_t \odot \tanh(c_t)$$

where

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c_t}$$

Where the $W\_$ and $b\_$ values are the weights and biases of their associated single layer neural network, $[x_t h_{t-1}]$ is a concatenation, and $\odot$ is pointwise-multiplication.

The output of the vision model is passed through two of these LSTM units to generate the final prediction. The reason for using two of these units was because hyperparameter tuning showed that two units allowed the system to acheive the best performance.

IV. Experiments

A. Experimental Setup

The dataset that was used for training was MNIST. The image size that was used 28 X 28, and the datasets were obtained from torchvision.datasets. The training set consisted of 60000 images, and the test set contained 10000 images.

The first phase of GlanceAgain are the transformations. The rotation angle and scaling factors were randomly chosen for each input, however they were set to be within a range. Specifically, the angles of rotation were allowed to be between $-30°$ and $30°$, and the scaling factor was chosen to be between 0.5 and 2. The angle range was not allowed to be very large because excessive rotation may cause an MNIST digit to not longer resemble the ground truth label for that digit. For example, an image of a seven rotated $180°$ would no longer resemble 7. As stated previously, the resulting batch of images were passed through the original vision model (after each image was flattened).

The fully-connected neural network which constituted the original model consisted of two linear layers where the output of the first layer was activated by ReLU, and the activation function of the last layer was LogSoftMax. The first layer had 100 neurons. As stated previously, the last layer was replaced in order to have it's output shape be compatible for input to the subsequent LSTM units. It was replaced to have 784 neurons (28 x 28 = 784), and its LogSoftmax activation function was replaced with a ReLU activation.

The LSTM layer simply used a torch.nn.LSTM implementation and consisted of a hidden size of 64 units. The whole model was optimized with an Adam optimizer, and it was allowed to run for 15 epochs. For the FGSM attack, $\epsilon = 0.3$ was used.

Along with the GlanceAgain model, the original vision model was also trained. The original vision model was trained normally (which we refer to as "NormalNet" in the following section) and adversarially (referred to as "AdvNet"). Hence, including GlanceAgain, there were three models in total.

B. Results and Discussion

All of the results are from un-perturbed or perturbed versions of the training data.

First, the accuracy of the models on the un-perturbed data was determined. These results are summarized in Table 1.

TABLE I
ACCURACY OF MODELS ON UNPERTURBED DATA

Accuracy (%)		
NormalNet	***AdvNet***	***GlanceAgain***
97.1	97.3	98.2

Although it was not the focus of this paper, it is interesting that the testing accuracy of GlanceAgain was higher than that of NormalNet and AdvNet for the unperturbed data. It may be the case that the use of sequences of images, as well as a recurrent component, improved the accuracy of the model.

After training all of the models, FGSM was used to generate adversarial samples from NormalNet. The resulting performance of NormalNet, AdvNet and GlanceAgain are summarized in Table 2.

TABLE II
ACCURACY OF MODELS ON ADVERSARIAL SAMPLES CRAFTED FOR NORMALNET

Accuracy (%)		
NormalNet	***AdvNet***	***GlanceAgain***
6.7	94.4	91.2

As table 2 indicates, an FGSM attack against NormalNet results in a catastrophic drop in accuracy, however, this is not

the case for GlanceAgain. The adversarially trained model, AdvNet, still performs best, but GlanceAgain's accuracy is relatively close to that of AdvNet.

Finally, NormalNet and GlanceAgain were tested against adversarially trained samples specifically crafted to attack these models themselves. The results are summarized in Table 3.

TABLE III
ACCURACY OF MODELS ON THEIR RESPECTIVE CRAFTED ADVERSARIAL SAMPLES

Accuracy (%)		
NormalNet	***AdvNet***	***GlanceAgain***
6.7	87.1	52.1

In this case, AdvNet performed the best, but it should be noted that GlanceAgain, despite not being specifically trained against adversarial samples, did not suffer as high of a drop in accuracy as NormalNet. The use of input sequences and a recurrent component may have made it more difficult to generate successful adversarial samples.

V. FURTHER DISCUSSION

Although it is not as effective at resisting FGSM attack as adversarial training, GlanceAgain still achieves a similar accuracy to adversarial training when encountering transferred samples, as was shown in Table 2. Furthermore, unlike [3], GlanceAgain does not experience a drop in accuracy of the underlying vision model. Results from Table 1 showed that the accuracy of GlanceAgain was indeed higher than the accuracy of the original model it was built from.

It should also be noted that GlanceAgain could be modified to include other adversarial defense methods as transformations. As stated previously, the output of HGD could be used as a transformed sample in GlanceAgain. It may even be possible to include transformations of adversarially perturbed samples in the transformation batch, essentially doing adversarial training. The result of using adversarial defense methods as transformations is that the resulting performance of GlanceAgain could be higher than that of each of the individual adversarial defense methods on their own. The adaptability of GlanceAgain to use different types of transformations is a significant benefit. Also, from Table 1, it may be the case that using sequences of inputs and a recurrent structure may improve the performance of many algorithms, however, this claim requires further experimentation.

However, there are several limitations to the study. First, as mentioned previously, the underlying vision model is not a CNN. This is a significant drawback since most modern day vision models are built with CNN's. Hence, in order to be more applicable to modern vision models, GlanceAgain should be tested against CNN-based models. Secondly, GlanceAgain should also be tested on more complex datasets like ImageNet. ImageNet has a wider variety of images and image classifications, and it has coloured images as well, thus being more realistic than grayscale MNIST images. Thirdly, GlanceAgain may work more effectively if the model learned to choose the number and type of transformations to conduct. Hence, as a next step, evolutionary strategies or reinforcement learning could be used to choose the best transformations to do. Furthermore, and most importantly, the drop in accuracy from the FGSM attack is still siginificant. In order to improve the model, more types of transformations could be employed as [3] did with random padding and resizing. Finally, the effectiveness of GlanceAgain against other adversarial attacks needs to be analyzed. FGSM is merely one of many adversarial attacks that exist.

VI. CONCLUSION

Overall, the method proposed in this paper, GlanceAgain, shows the potential to be a viable adversarial defense method. It achieves an accuracy close to that of adversarial training on adversarial samples generated from an FGSM attack on the original (target) vision model. Also, unlike the original model alone, the GlanceAgain model is less impervious to FGSM attacks crafted against itself. Further work needs to be conducted to increase the model's sophistication and compare its effectiveness against more complex attacks. However, due to this method's similarity with the method in [3], and with the added benefit of incorporating more adversarial defense techniques as transformations, this method may be able to produce more notable results.

REFERENCES

[1] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on Deep Learning Visual Classification," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Apr. 2018.

[2] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 4, pp. 677–691, 2014.

[3] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," presented at the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, Apr. 30-May 3, 2018, Paper 352.

[4] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," presented at the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015.

[5] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," presented at the 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, Apr. 30-May 3, 2018, Paper 835.

[6] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided Denoiser," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1778–1787, Jun. 2018.

[7] M. Jagielski, S. Wu, A. Oprea, J. Ullman, and R. Geambasu, "How to combine membership-inference attacks on multiple updated models," arXiv.org, 12-May-2022. [Online]. Available: https://doi.org/10.48550/arXiv.2205.06369. [Accessed: 08-Aug-2022].

[8] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of Recurrent Neural Networks: LSTM cells and network architectures," Neural Computation, vol. 31, no. 7, pp. 1235–1270, Jul. 2019.

[9] Jeff Orchard, *Long Short-Term Memory (LSTM)*. (2021). Accessed: Aug. 7, 2022. [Online Video]. Available: https://www.youtube.com/watch?v=2nt284h6sYw&list=PLzB6s7uqJBd4xvNEk9XluyZ6l_cf4QLnp&index=24