Replacing FC layers in a CNN could improve robustness against adversarial attacks

Itamar F. Salazar-Reque[1], Gonzalo Otazu Aldana[2], Samuel Huamán Bustamante[1]

[1]Instituto Nacional de Investigación y Capacitación de Telecomunicaciones, Universidad Nacional de Ingeniería, Lima-Perú

[2]Department of Biomedical Science, College of Osteopathic Medicine, New York Institute of Technology, Old Westbury

UNIVERSIDAD NACIONAL DE INGENIERÍA
INSTITUTO NACIONAL DE INVESTIGACIÓN Y CAPACITACIÓN DE TELECOMUNICACIONES

Introduction

Many classification techniques, including convolutional neural networks(CNN), have shown instabilities when they are exposed to “adversarial examples” [1-3]. In image classification, an adversarial example is an image having imperceptible perturbations to a human observer that change the prediction of the classifier (e.g. a CNN) [2]. Given an image sample ($\mathbf{x}$), the adversarial perturbation ($\mathbf{r}$) is such that:

$$\hat{k}(\mathbf{x}+\mathbf{r}) \neq \hat{k}(\mathbf{x})$$

where, $\hat{k}(\mathbf{x})$ is the CNN predicted label for any given input image $\mathbf{x}$. An example of an adversarial example ($\mathbf{x}+\mathbf{r}$) can be seen in Figure 01.

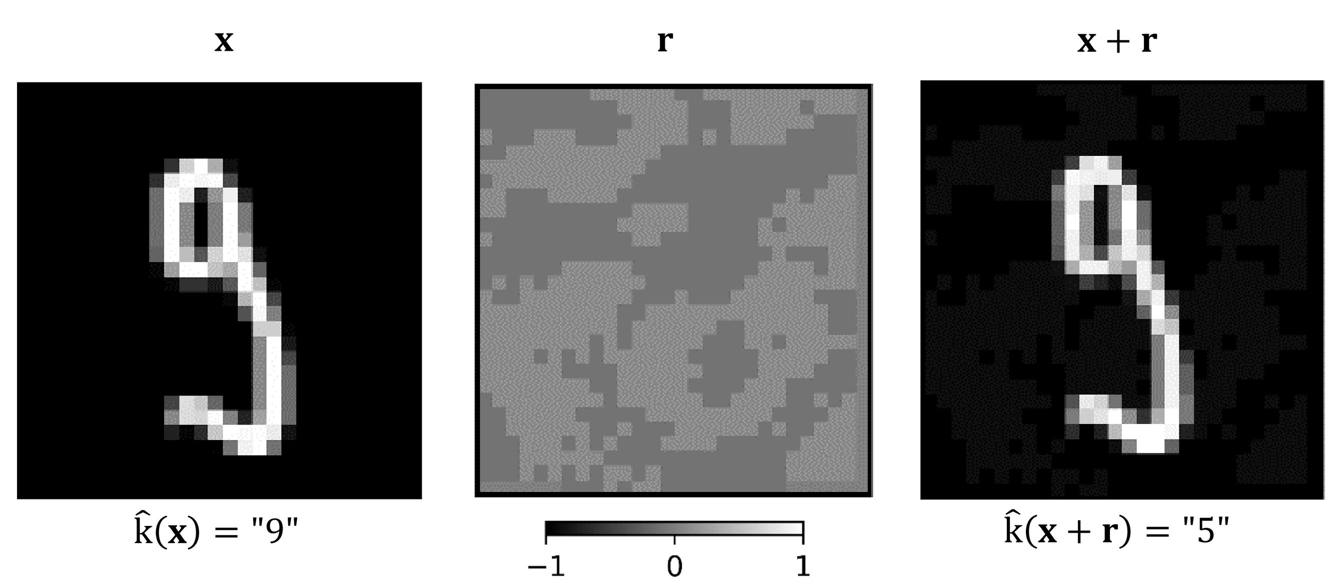

Figure 01. Adding an adversarial perturbation ($\mathbf{r}$) to an input image ($\mathbf{x}$) creates an adversarial image ($\mathbf{x}+\mathbf{r}$) for which the estimated label $\hat{k}(\mathbf{x}+\mathbf{r})$ is now wrong.

We can decrease CNN accuracy if we increase the amplitude of the perturbation (ε), as can be seen in Figure 02. In that figure, accuracy is computed for training and testing data on the MNIST dataset for a CNN with 2 convolutional layers.

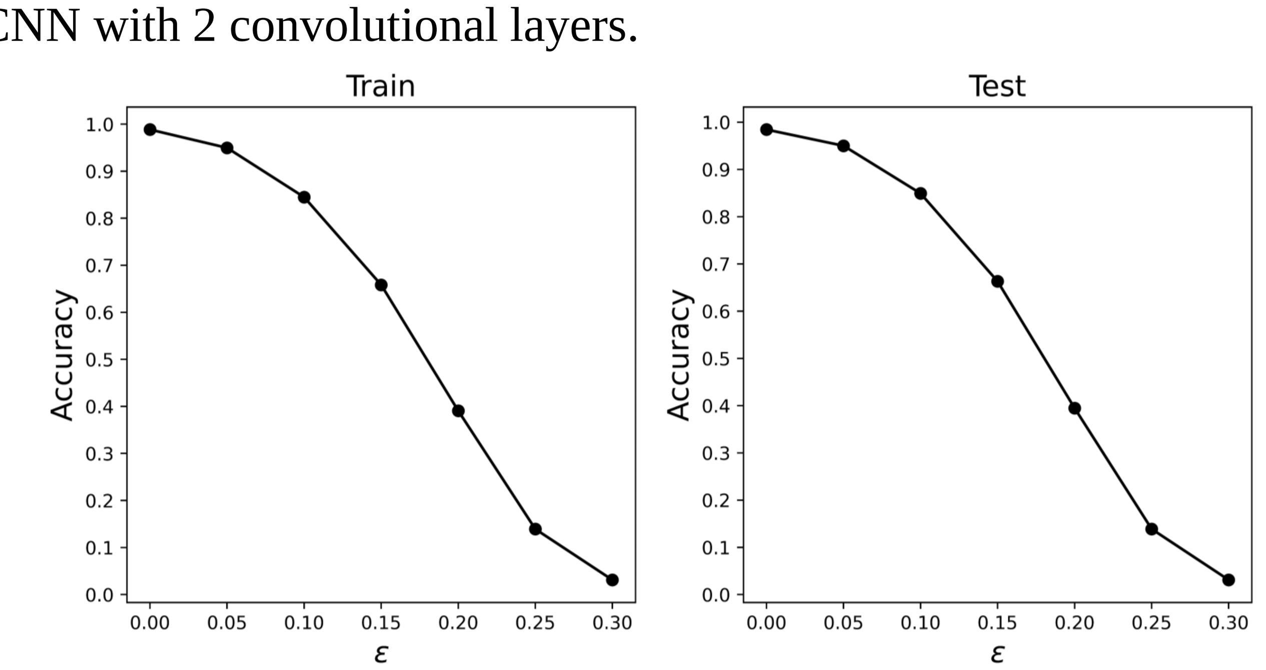

Figure 02. Adversarial examples classification accuracy for a CNN trained with MNIST dataset. Accuracy is shown for adversarial examples created from training and test sets and for different amplitude values of the perturbation ε.

Procedure

DICTIONARY

We first trained a 2-layer CNN with FCs on top using MNIST dataset. We compute activations from the second convolutional layer per class and pool them to obtain a 50-dimensional vector per image $\mathbf{V}(\mathbf{x})$.
We use the average of these vectors per class as representative atoms $\mathbf{A}_\mathrm{k}$ for class k to create a dictionary $\mathbf{\Phi} \in \mathbb{R}^{\mathrm{N}\times\mathrm{K}}$ where N = 50 and K = 10, such that:

$$\mathbf{\Phi} = \begin{bmatrix} | & | & & | \\ \mathbf{A_0} & \mathbf{A_1} & \dots & \mathbf{A_9} \\ | & | & & | \end{bmatrix}$$

ADVERSARIAL PERTURBATIONS

We compute adversarial perturbations per image using Fast Gradient Sign Method [1].

$$\mathbf{r} = \varepsilon \times \operatorname{sign}\left(\nabla_{\mathbf{x}} \mathcal{J}(\boldsymbol{w}, \mathbf{x}, k(\mathbf{x}))\right)$$

where $\mathbf{w}$ represent the network parameters and $\mathcal{J}$ is the loss function.

CLASIFICATION USING KNN AND CPA

For each adversarial example ($\mathbf{x}+\mathbf{r}$) we compute the 50-dimensional vector $\mathbf{V}(\mathbf{x}+\mathbf{r})$ and estimate the correct label using the dictionary $\mathbf{\Phi}$. We used two algorithms to solve this problem: KNN (k-nearest neighbors) and CPA(corrected projection algorithm).

For KNN, we estimate the 3 nearest atoms from the dictionary $\mathbf{\Phi}$ to $\mathbf{V}(\mathbf{x}+\mathbf{r})$, then the label would be:

$$\hat{k}(\mathbf{x}+\mathbf{r}) = \min_{k} \mathrm{d}_{jk}$$

$$\mathrm{d}_{\mathrm{jk}} = \left\| \mathbf{V}(\mathbf{x}_\mathrm{j}+\mathbf{r}) - \mathbf{A}_\mathrm{k} \right\|_2$$

In the case we use $\mathbf{L}$ adversarial examples belonging to the same class, the estimated label would be:

$$\hat{k}(\mathbf{x}+\mathbf{r}) = \operatorname*{argmin}_{k} \sum_{j=1}^{L} \mathrm{d}_{jk}$$

For CPA, we need to reformulate the problem as a dictionary problem, such that:

$$\mathbf{V}(\mathbf{x}+\mathbf{r}) = \mathbf{\Phi}\boldsymbol{\theta}$$

Where, $\boldsymbol{\theta}$ is the vector of “presence parameters”. Then our objective is to calculate $\boldsymbol{\theta}$ and estimate the label using:

$$\hat{k}(\mathbf{x}+\mathbf{r}) = \operatorname*{argmax}_{k} \theta_\mathrm{k}$$

CPA is an algorithm that is robust to novel distractors and handles high dimensional atoms [4]. In this case, the perturbation added is not part of the dictionary of digit prototypes so CPA will ignore it. The algorithm obtains the presence parameters $\boldsymbol{\theta}$ using:

$$\boldsymbol{\theta} = \left(\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T} + \lambda\mathbf{I}\right)^{-1}\mathbf{\Phi}^\mathbf{T}\mathbf{V}(\mathbf{x}+\mathbf{r})$$

which is solved iteratively for $\mathbf{L}$ adversarial examples using a Kalman-filter type of algorithm [4].

Results

With no perturbations (ε=0) and L =1, KNN and CPA obtain inferior results compared to FC. For L = 4 results were similar.
For an amplitude perturbation of ε = 0.25 a CNN with KNN or CPA layer at the end performs at least 55% of accuracy whereas the standard network (with FC at the end) collapses to 15 percent (see Figure 03).

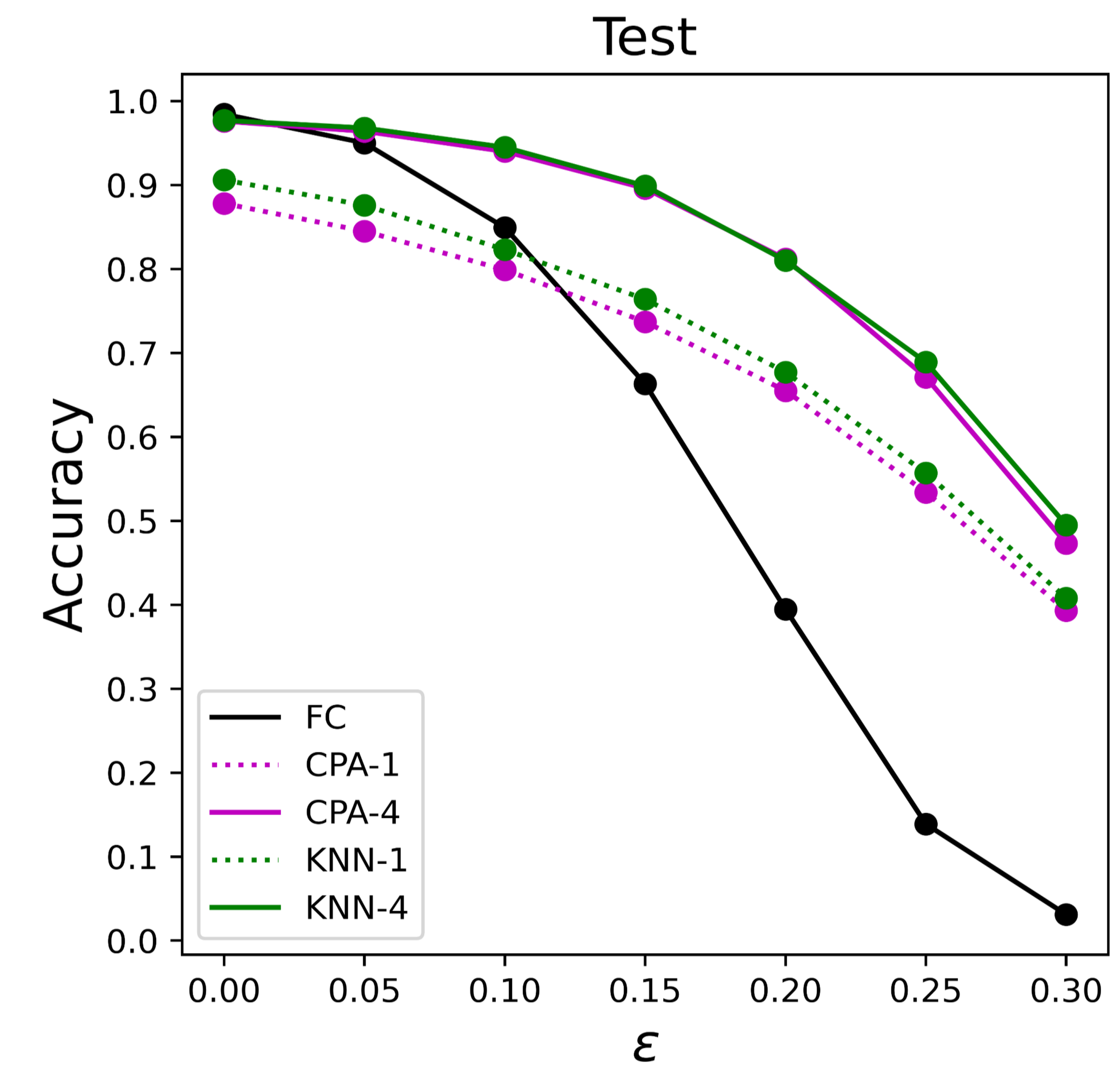

Figure 03. Adversarial examples classification accuracy for a trained CNN after changing FC layer for KNN layer or CPA layer. Accuracy is shown for adversarial examples created from test set and for different amplitude values of the perturbation ε.

Conclusions

- We explored the effect of KNN and CPA algorithms as classifiers of CNN visual representations.
- Replacing the last fully connected segment for an algorithm that operates by proximity could improves CNN robustness against adversarial perturbation.
- By increasing the perturbation amplitude CPA and KNN show better accuracy in the classification of low dimensional activations generated by adversarial inputs, this is evident for ε>0.15.

Future work

To increase activations dimensionality by making the architecture deeper and train it with other datasets like CIFAR or ImageNet.

References

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–11, 2015.

[2] S. M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-December, pp. 2574–2582, 2016, doi: 10.1109/CVPR.2016.282.

[3] C. Szegedy et al., “Intriguing properties of neural networks,” 2nd Int. Conf. Learn. Represent. ICLR 2014 - Conf. Track Proc., pp. 1–10, 2014.

[4] G. H. Otazu, “Robust method for finding sparse solutions to linear inverse problems using an L2 regularization,” pp. 1–13, 2017, [Online]. Available: http://arxiv.org/abs/1701.00573

Acknowledgements

The authors would like to thank INICTEL-UNI and NYIT College of Osteopathic Medicine for the support.

NYIT College of Osteopathic Medicine

Itamar Franco Salazar Reque | isalazarr@uni.pe | Github: itamarfrancosr | Twitter: @ItamarSalazar | Linkedin: itamarsalazar