Adversarial Training of Neural Networks for Cryptography Applications

Chandrasekhar S, Nagabhushan S N, Sandesh Rao M
Machine Learning Project
Indian Institute of Science

April 27, 2019

Motivation

- Can neural networks be trained adverserially to encrypt and decrypt data?
- Alice and Bob try to communicate securely without Eve being able to decipher the data.

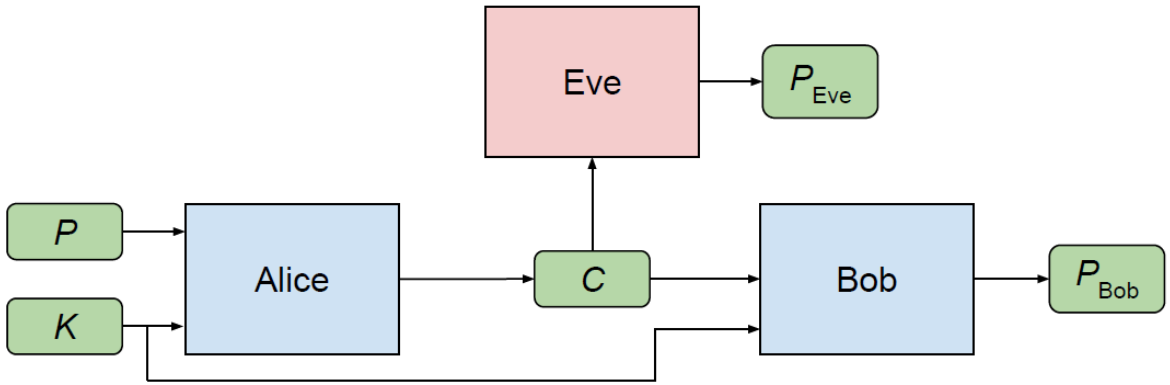


Figure: Basic Structure. Courtesy [1]

Introduction-Loss functions

- The paper uses L1 loss function -

$$L_E(\theta_A, \theta_E, P, K) = d(P, E(\theta_E, A(\theta_A, P, K))) \tag{1}$$

$$d(P, \hat{P}) = \sum_{i=1}^{N} |P_i - \hat{P}_i| \tag{2}$$

$$L_E(\theta_A, \theta_E) = \mathbb{E}_{P,K}(L_E(\theta_A, \theta_E, P, K)) \tag{3}$$

$$\theta_E^*(\theta_A) = \arg\min_{\theta_E} L_E(\theta_A, \theta_E) \tag{4}$$

$$L_B(\theta_A, \theta_B, P, K) = d(P, B(\theta_B, A(\theta_A, P, K), K)) \tag{5}$$

$$L_B(\theta_A, \theta_B) = \mathbb{E}_{P,K}(L_B(\theta_A, \theta_B, P, K)) \tag{6}$$

$$L_{AB}(\theta_A, \theta_B) = L_B(\theta_A, \theta_B) - L_E(\theta_A, \theta_E^*(\theta_A)) \tag{7}$$

$$(\theta_A^*, \theta_B^*) = \arg \min_{\theta_A, \theta_B} L_{AB}(\theta_A, \theta_B) \tag{8}$$

- Instead of the difference between the L1-losses between Bob and eve, they have also proposed the use of the following L1-loss, we call this the "Modified L1 Loss" and the above mentioned Loss as "Simple L1 Loss"

$$L_{AB}(\theta_A, \theta_B) = L_B(\theta_A, \theta_B) + \frac{(N/2 - L_E(\theta_A, \theta_E^*(\theta_A)))^2}{(N/2)^2} \tag{9}$$

Recreation of results from Paper

- We used batch size of 4096 randomly generated data per step of Adam optimizer.

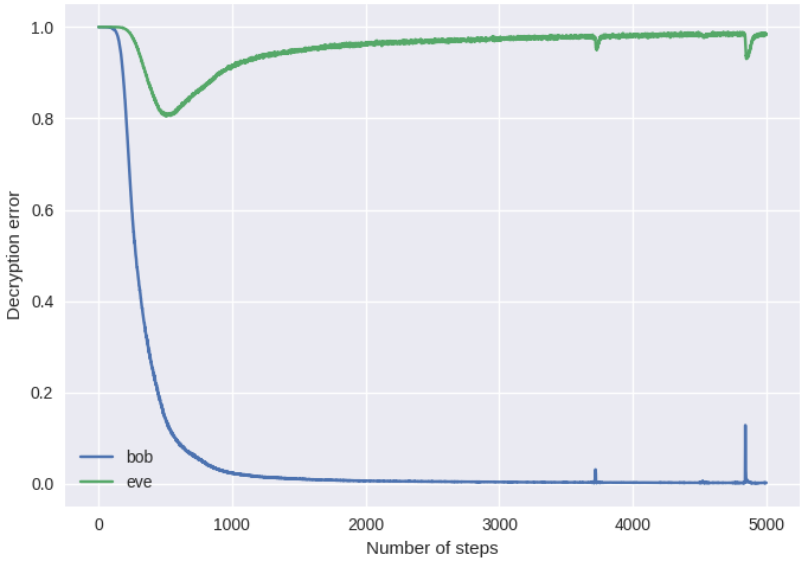


Figure: Simple L1 loss

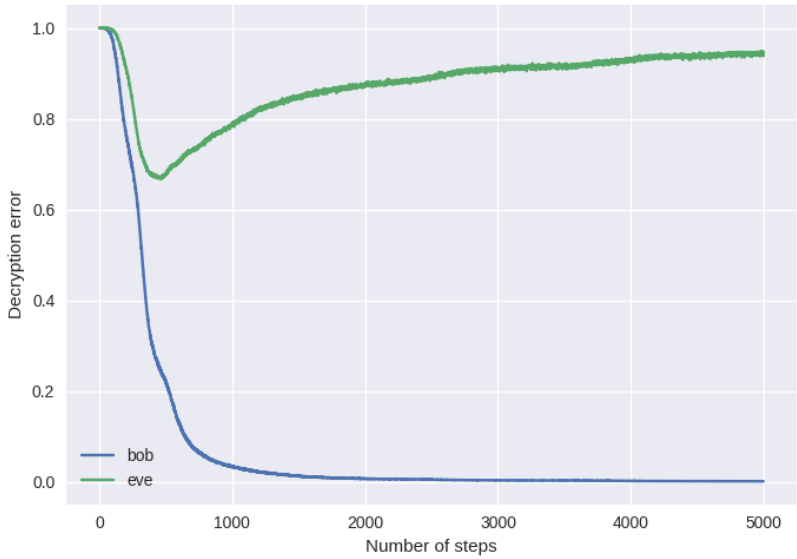
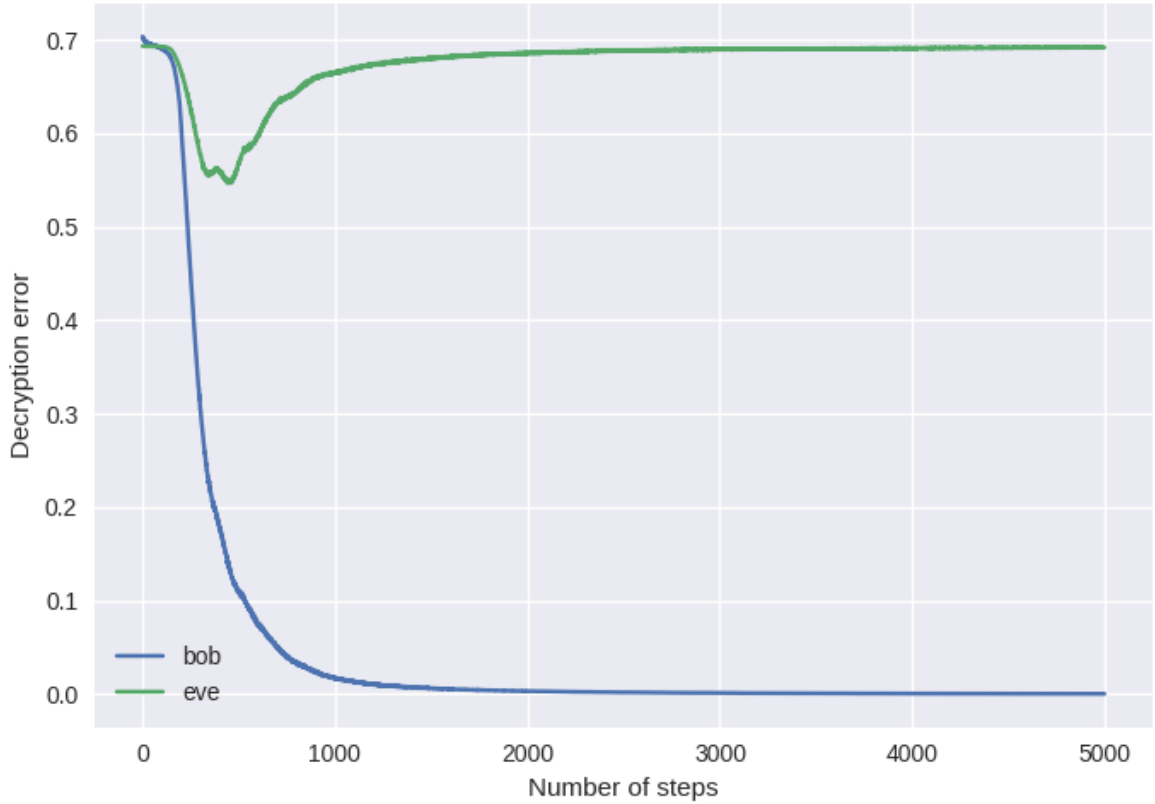


Figure: Modified L1 loss

Ablation Study 1: BCE Loss function

- When we minimize using simple and modified L1 loss functions, we observed that Eve predicts on an average 8 bits wrong and 8 bits correct.
- Instead when we used BCE loss function and when Alice had trained completely, we observed that Eve predicted close to probability= 0.5 for each bit.



Ablation Study 2: Eve trained longer

- Alice and Bob should be able to communicate securely and Eve should not be able to decrypt the messages even if she has a large collection of message and cipher-text.
- Once Alice and Bob are fixed, if training of Eve is continued, can Eve decrypt the messages?

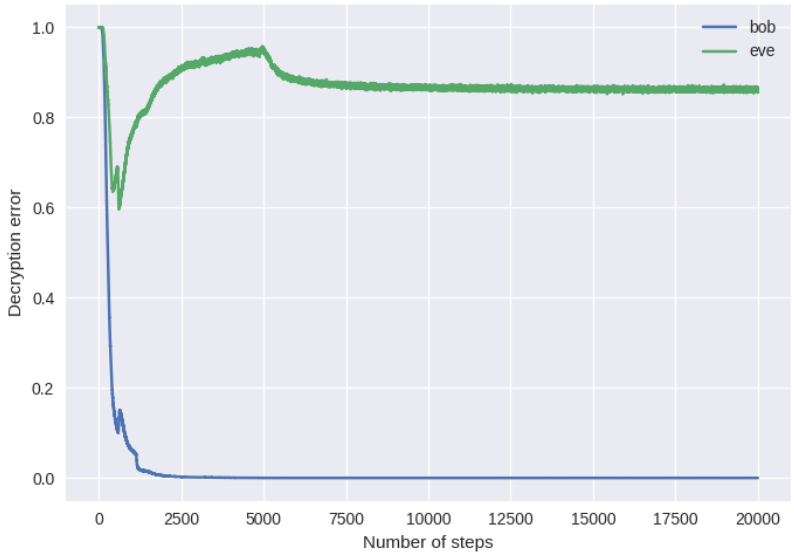


Figure: Simple L1 loss

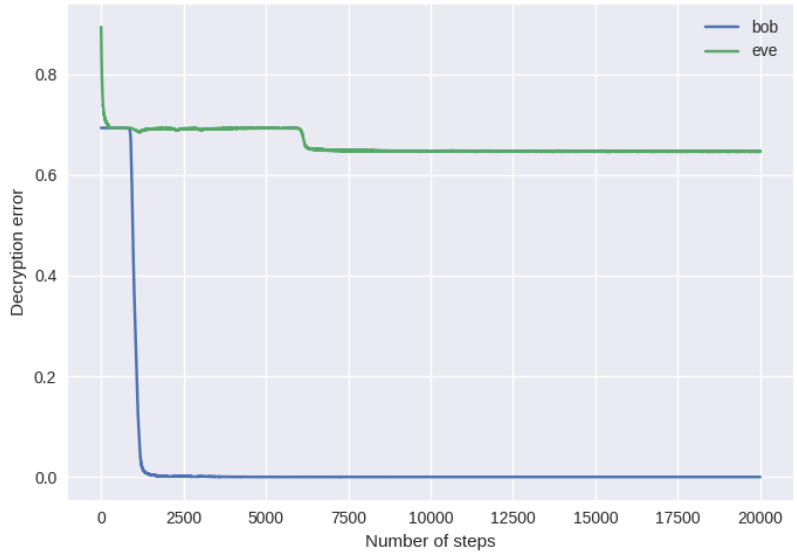


Figure: BCE Loss

Ablation Study 3: Shorter Keys

- In the paper, they use message and key to be of same length (16). But it is not practical to have a key as long as message. So, we reduced the key length and tested the performance.

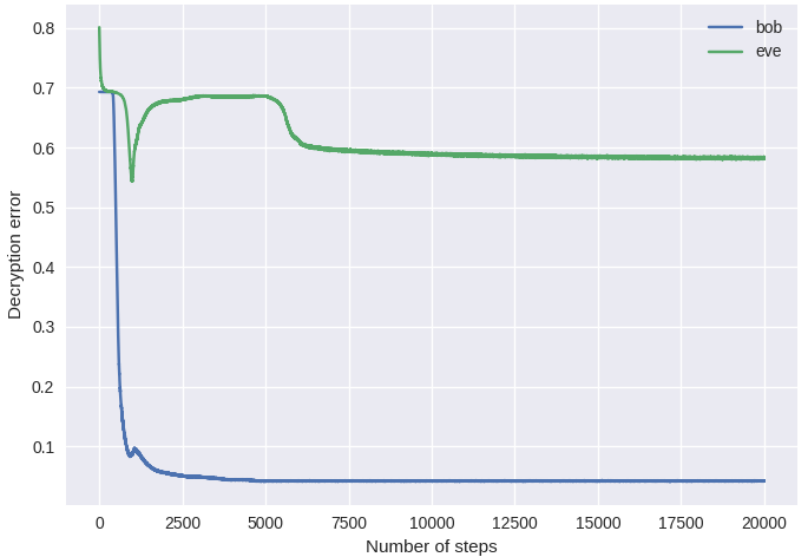


Figure: Key length 8

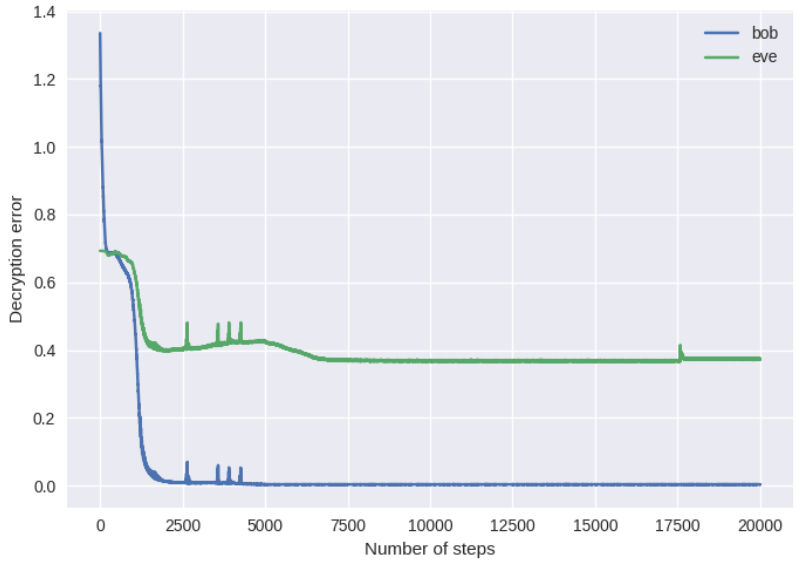


Figure: Key length 1

1: Binarizing Alice's output

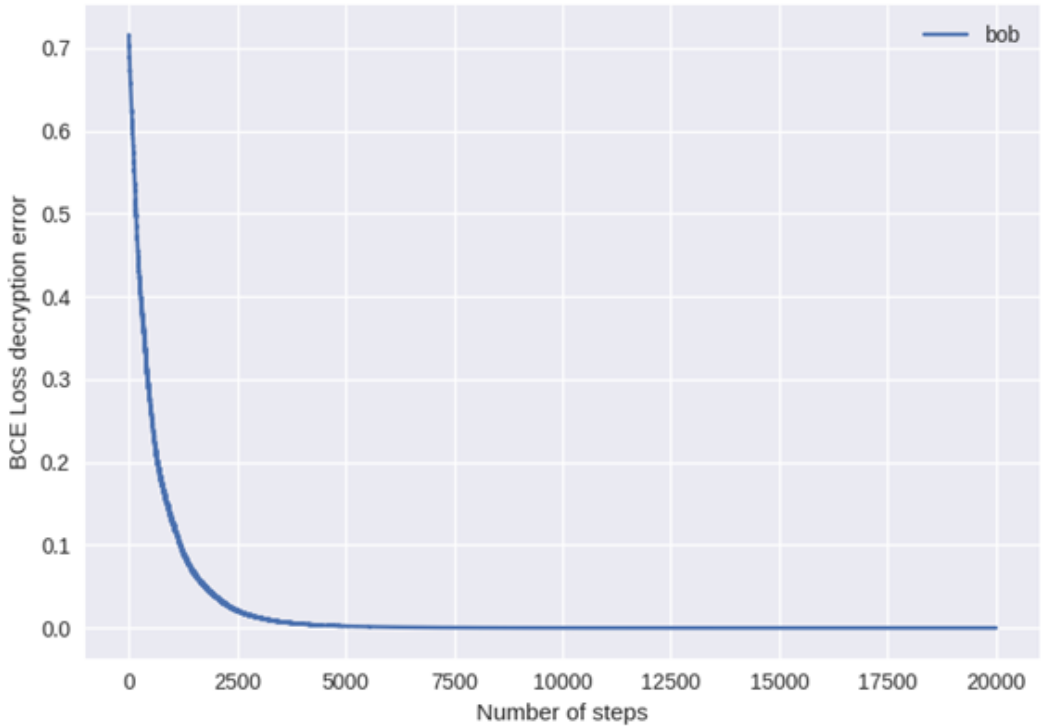
- In the current implementation of the paper, Alice's output are float (real) values.
- When we want to transmit an encrypted message, we would have to transmit it in bits
- Add a binarizing (thresholding) layer to Alice's output to get binary ciphertext.

2: Impersonating Alice

- Alice and Bob have been trained till convergence
- Suppose Eve provides input message to Alice (a fixed key is used by Alice) and the ciphertext generated is sent to Bob
- Eve has blackbox access to Bob
- We train on the key to recover the original key used by Alice
- Gathering many such instances of data, we can train another network that acts like Alice

3: Extension to Error Correcting Codes

- Suppose Eve is not able to decrypt the messages, but can influence the transmission channel and flip some of the bits.
- Add some redundant bits, ciphertext will be longer than message
- If Alice's output is real values, then Alice can embed a lot of information in the output
- Following is the result of sending 4 bits with 3 redundant bits ((7,4) Hamming code)



4: Digital Communication

- The job of the encoder is to map input messages taking M possible values to N real numbers
- The job of the decoder is to undo what the encoder has done
- In this setting, we ask the following question, if the encoder and the decoder blocks are replaced by neural networks and made to train to minimize the decoding errors, will the neural networks converge to good encoders and decoders.

Figure: Block diagram for analysis

THANK YOU!

Questions?

M. Abadi and D. G. Andersen, "Learning to protect communications with adversarial neural cryptography," *arXiv preprint arXiv:1610.06918*, 2016.

A. Anand, "neural-cryptography-tensorflow," https://github.com/ankeshanand/neural-cryptography-tensorflow, 2016, [Online; accessed March-2019].