

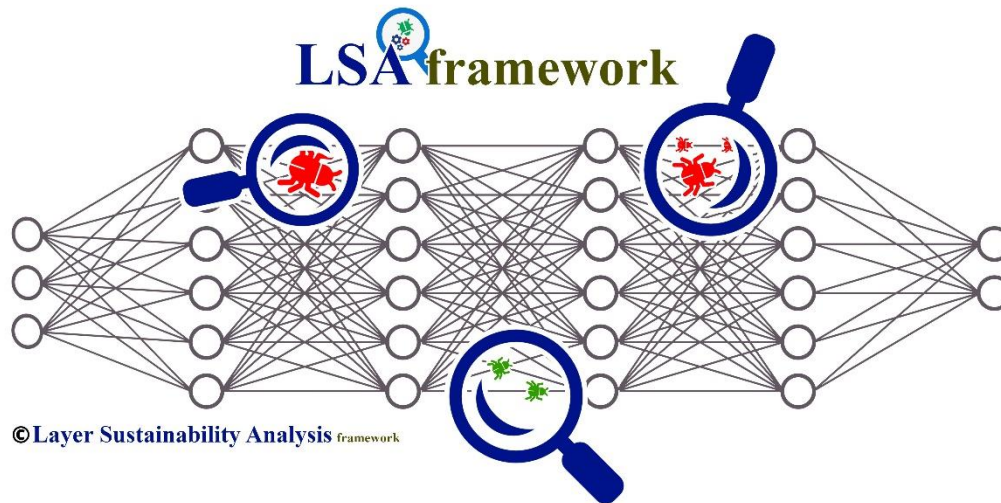


# Robustness of Deep Neural Networks against white-box adversarial attacks

Mohammad Khalooei , Mohammad Mehdi Homayounpour , Maryam Amirmazlaghani



- Review on prev. presentations
- Paper review
  - Layer-wise **R**egularized **A**dversarial **T**raining (AT-LR)  
using **L**ayers **S**ustainability **A**nalysis (LSA) framework



## • Attack

$$x^* = x + \epsilon \cdot \text{sign}(\nabla_x J(x, y))$$

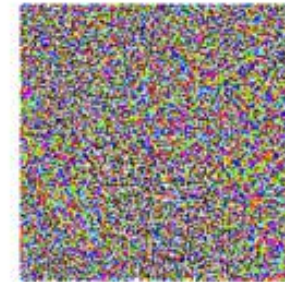
$$L_\infty \text{ norm } \|x^* - x\|_\infty \leq \epsilon$$

$$\arg \min_{\delta} \|\delta\|_p + c \cdot J(x + \delta, y) \\ \text{s.t. } x + \delta \in [0, 1]^n$$



$x$   
“panda”  
57.7% confidence

+ .007 ×



$\text{sign}(\nabla_x J(\theta, x, y))$   
“nematode”  
8.2% confidence

=



$x + \epsilon \text{sign}(\nabla_x J(\theta, x, y))$   
“gibbon”  
99.3 % confidence

## • Defense

- Data manipulation
- Architecture manipulation
- Loss manipulation

$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y) \sim \mathcal{D}} \left[ \max_{\delta \in \mathcal{S}} L(\theta, x + \delta, y) \right]$$



## • Adversarial Training (AT)

A method for learning robust deep networks

It is typically assumed to be **more expensive** than **traditional training**

constructing adversarial examples via a first-order method like projected gradient decent (PGD)

$$\min_{\theta} \sum_i \max_{\delta \in \Delta} \ell(f_{\theta}(x_i + \delta), y_i)$$

robust optimization problem

$$\Delta = \{\delta : \|\delta\|_{\infty} \leq \epsilon\} \quad \epsilon > 0$$

Madry et al. (2017)

$$\delta^* = \epsilon \cdot \text{sign}(\nabla_x \ell(f(x), y))$$

Goodfellow et al. (2014)

$(x_i, y_i)$  Dataset

$\ell$  Loss function

$f_{\theta}$  Network

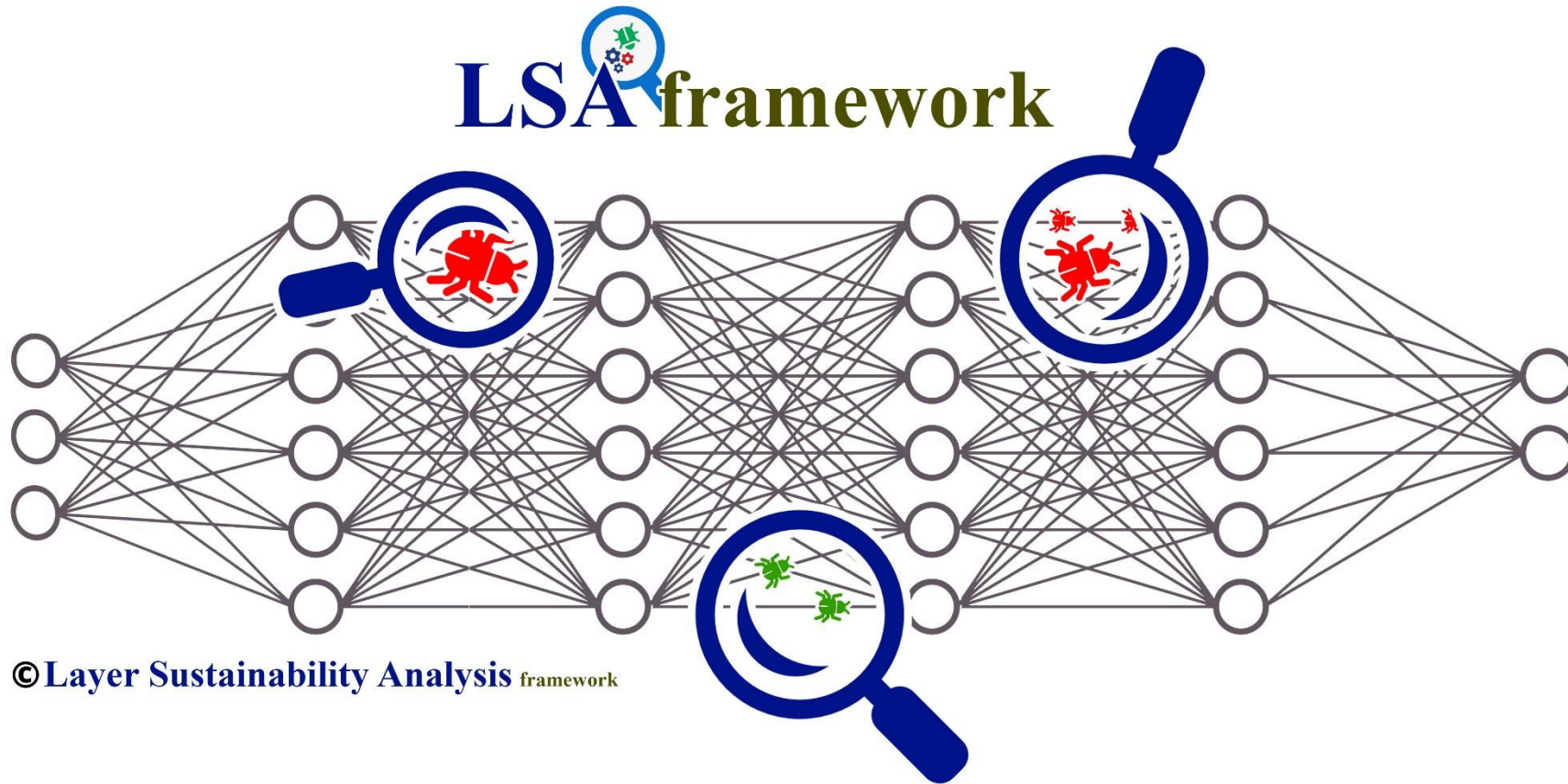
$\theta$  Parameters

$\Delta$  Threat model

A better approximation of the **inner maximization** is to take **multiple**



- Layer-wise Regularized Adversarial Training using **Layers Sustainability Analysis (LSA)** framework

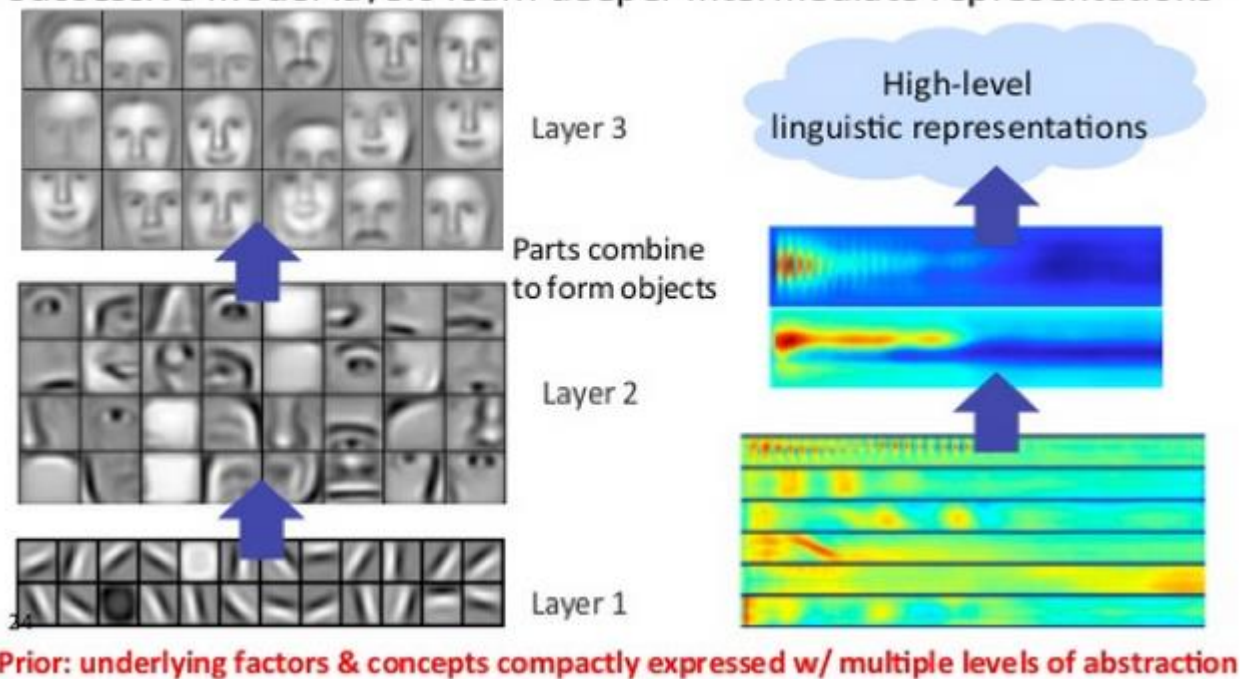


<https://github.com/khalooei/LSA>

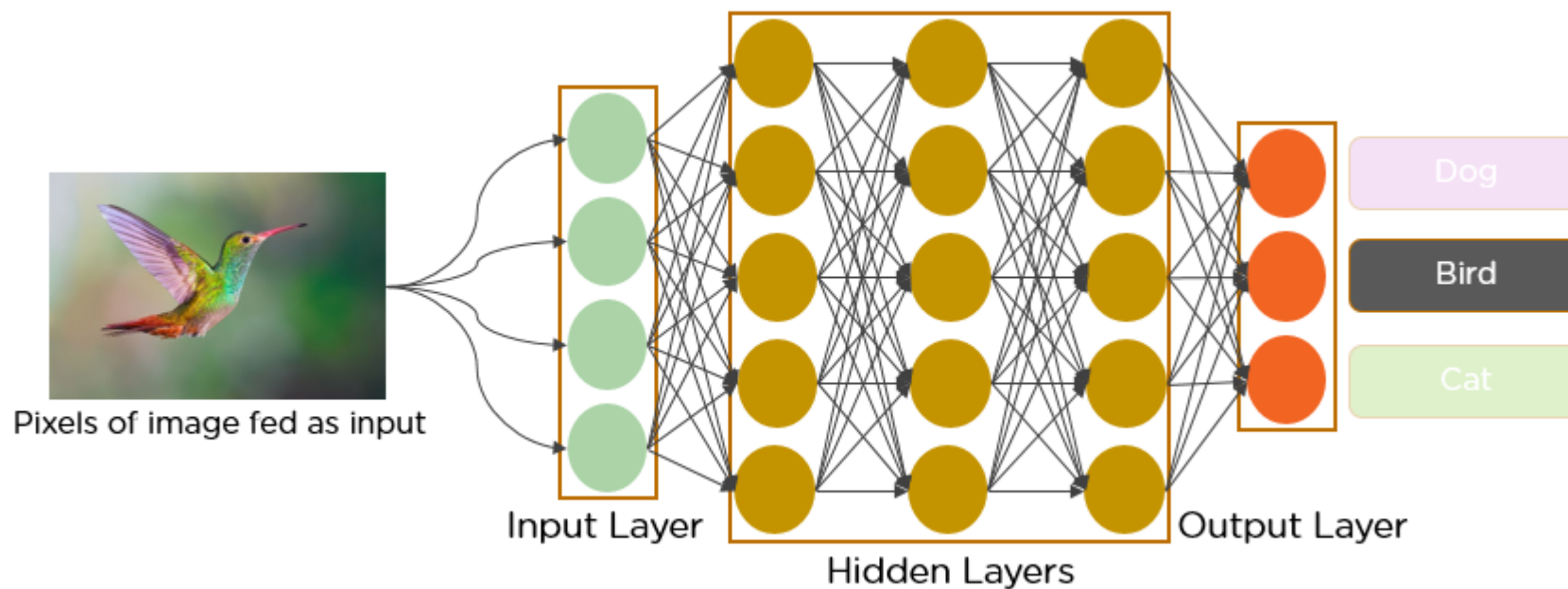




Successive model layers learn deeper intermediate representations



<https://wiki.pathmind.com/neural-network>





- Layer-wise Regularized Adversarial Training using **Layers Sustainability Analysis (LSA)** framework

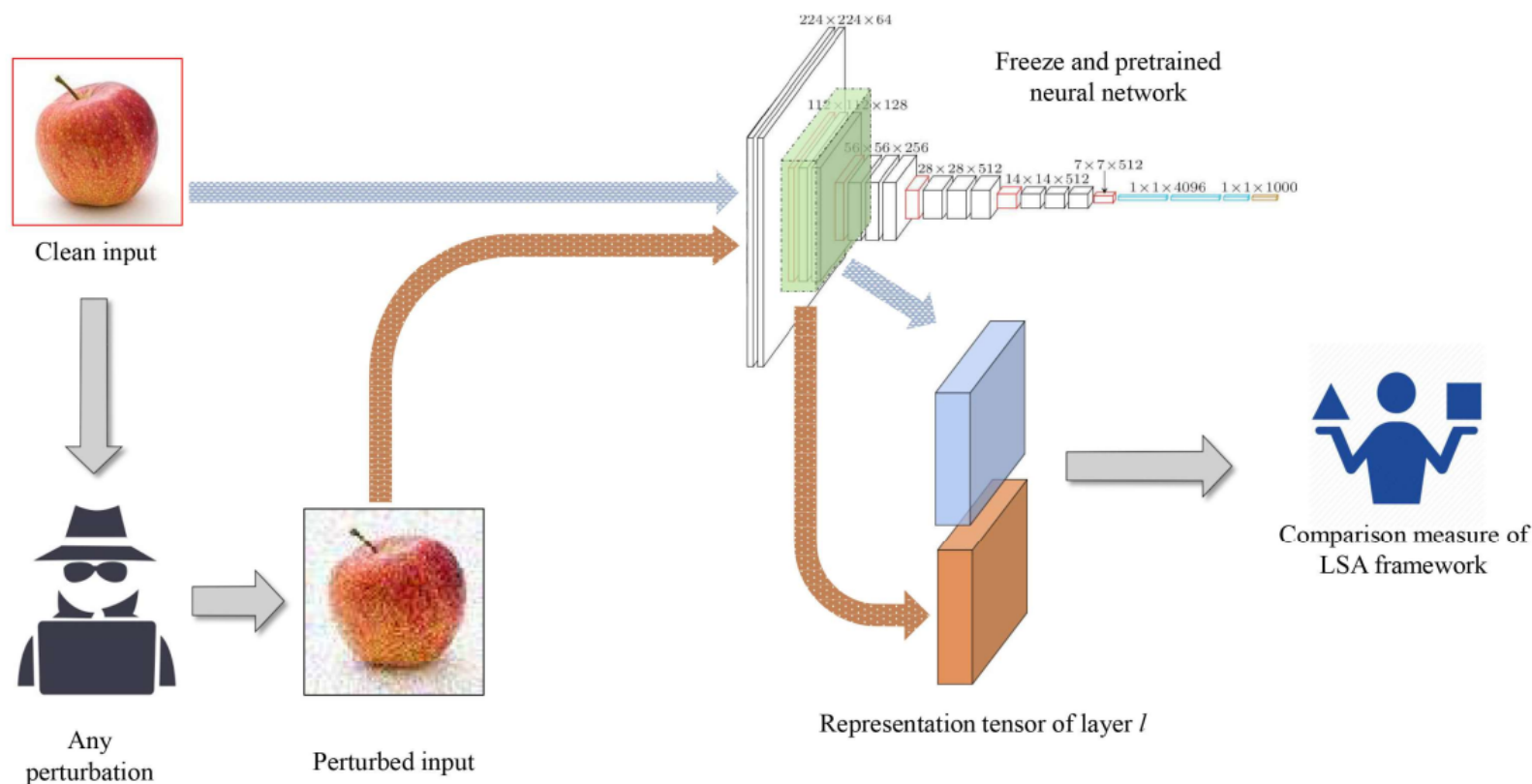


Fig. 1. Diagram of the Layer Sustainability Analysis (LSA) framework

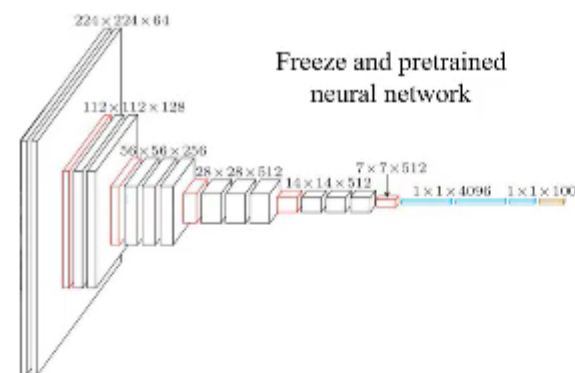




- Layer-wise Regularized Adversarial Training using **Layers Sustainability Analysis (LSA)** framework



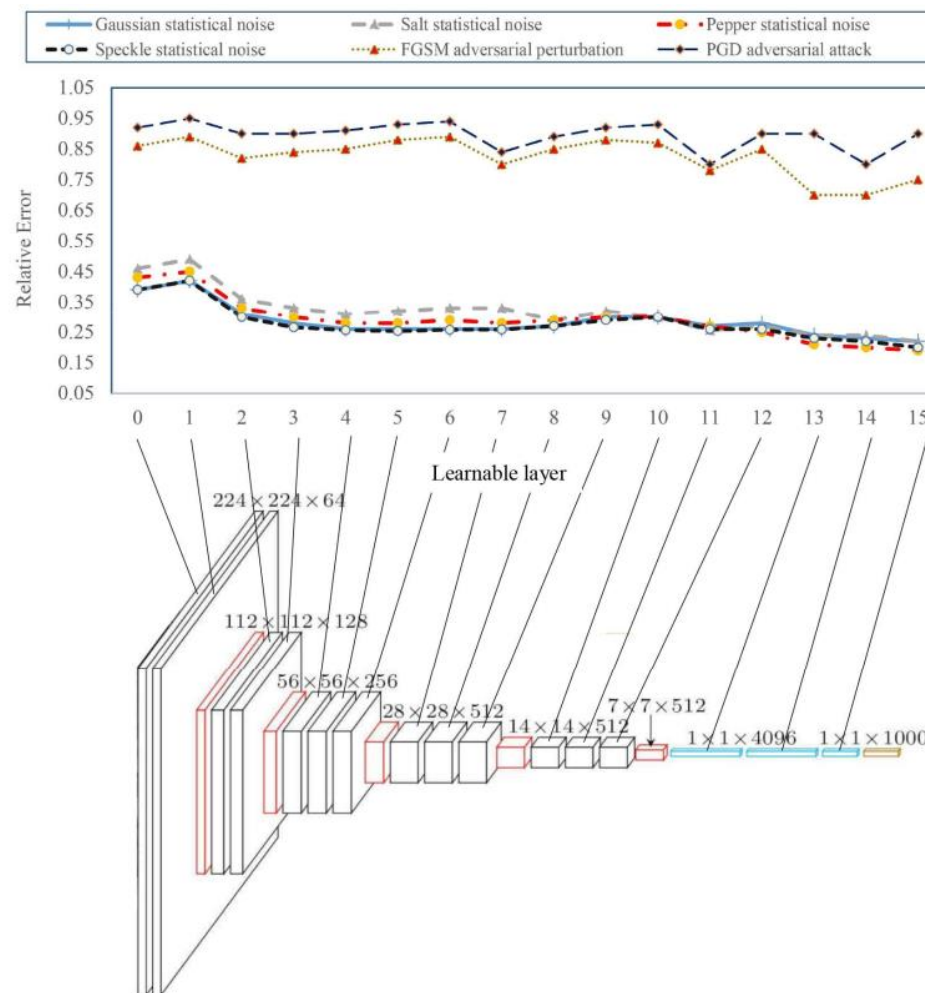
Clean input



Layer Sustainability Analysis (LSA) framework



- Layer-wise Regularized Adversarial Training using **Layers Sustainability Analysis (LSA)** framework



**Fig. 2.** Comparison measure values for corresponding layers of the VGG network in the proposed LSA framework



## Lipschitz condition

**Definition:** function  $f(t, y)$  satisfies a **Lipschitz condition** in the variable  $y$  on a set  $D \subset \mathbf{R}^2$  if a constant  $L > 0$  exists with

$$|f(t, y_1) - f(t, y_2)| \leq L |y_1 - y_2|,$$

whenever  $(t, y_1), (t, y_2)$  are in  $D$ .  $L$  is Lipschitz constant.

$$\|F(x_1) - F(x_2)\| \leq \psi \|x_1 - x_2\| \quad s.t. \ x_1, x_2 \in X.$$



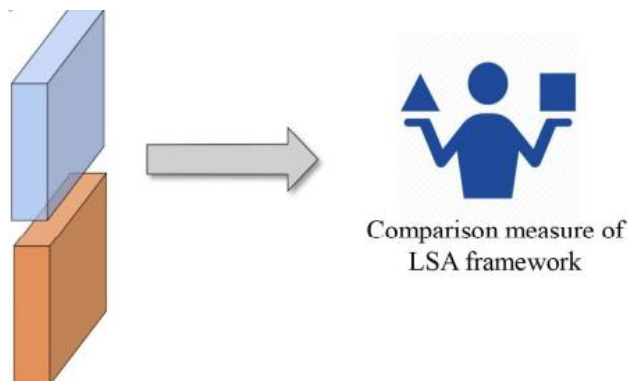
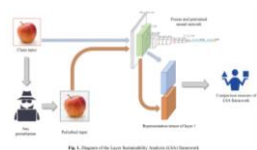
$$\|F(x_1) - F(x_2)\| \leq \psi \|x_1 - x_2\| \quad s.t. \ x_1, x_2 \in X.$$



$$\|\phi_l(x_1) - \phi_l(x_2)\| \leq \psi \|x_1 - x_2\|.$$



$$\|\phi_l(x) - \phi_l(\hat{x})\| \leq \psi \|x - \hat{x}\|.$$



$$CM(\phi_l(x), \phi_l(\hat{x})) = \frac{\|\phi_l(x) - \phi_l(\hat{x})\|_F}{\|\phi_l(x)\|_F},$$

$$\mu = \frac{1}{M \times Ly} \sum_{m=0}^{M-1} \sum_{l=0}^{Ly-1} CM(\phi_l(x_m), \phi_l(\hat{x}_m)),$$

$$\sigma = \sqrt{\frac{1}{M \times Ly} \sum_{m=0}^{M-1} \sum_{l=0}^{Ly-1} (CM(\phi_l(x_m), \phi_l(\hat{x}_m)) - \mu)^2},$$



**Algorithm 1.** Algorithm to find the most vulnerable layers in the layer sustainability analysis (LSA) framework

---

**Algorithm 1 .** Algorithm to find the most vulnerable layers in the layer sustainability analysis (LSA) framework

---

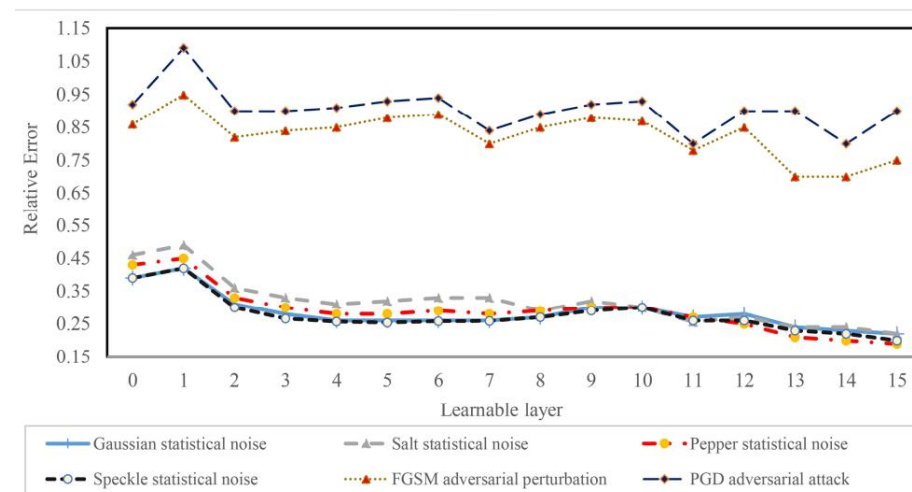
**Input:** Output representation tensors  $\phi_l(x)$  and  $\phi_l(\hat{x})$  of layer  $l$  for clean input  $x$  and the corresponding perturbed sample  $\hat{x}$

**Output:** list of most vulnerable layers

**Algorithm steps:**

for trained model  $m$ , constant  $\eta$ , average  $\mu$  and standard deviation  $\sigma$  as calculated in equation (12).

1. LSA\_MVL\_list = []
2. for  $l$  in range(0,  $L_y$ )
3.     if( $CM(\phi_l(x), \phi_l(\hat{x})) - \mu$ ) >  $\eta \sigma$
4.         LSA\_MVL\_list.append( $l$ )
5. LSA\_MVL\_list = sort(LSA\_MVL\_list)







**Algorithm 2.** Layer sustainability analysis (LSA) framework's algorithm

---

**Algorithm 2** Layer Sustainability Analysis (LSA) framework

---

**Input:** model  $m$ , train data  $D_{train}$ , test data  $D_{test}$ , attack method and its parameters

**Output:** list of vulnerable layers

**Algorithm steps:**

1. Standard training of model  $m$  using  $D_{train}$  samples
  2. Execute attack on the trained model  $m$  and obtain adversarial examples by perturbing  $D_{test}$  samples
  3. Run Algorithm 1 and tune the proper cut-off threshold  $\eta$  of Algorithm 1 to find out the MVL list
  4. Return the MVL list of Algorithm 1
-



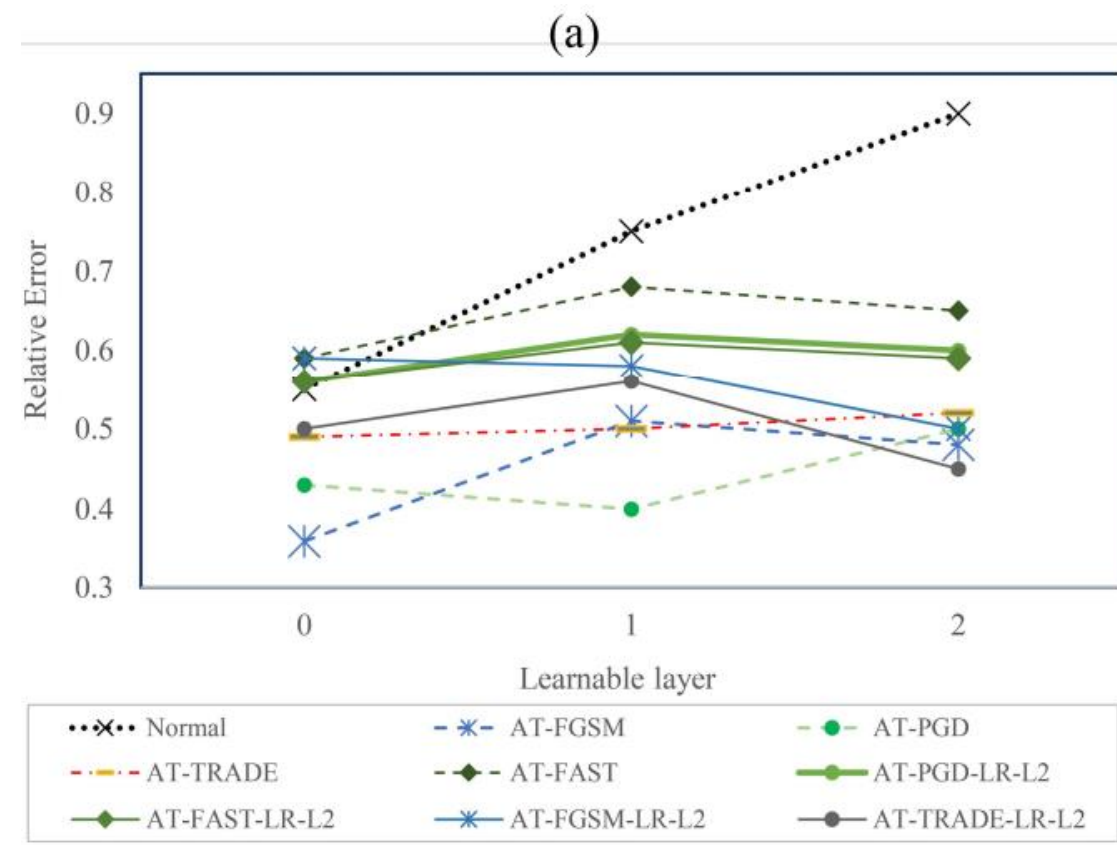
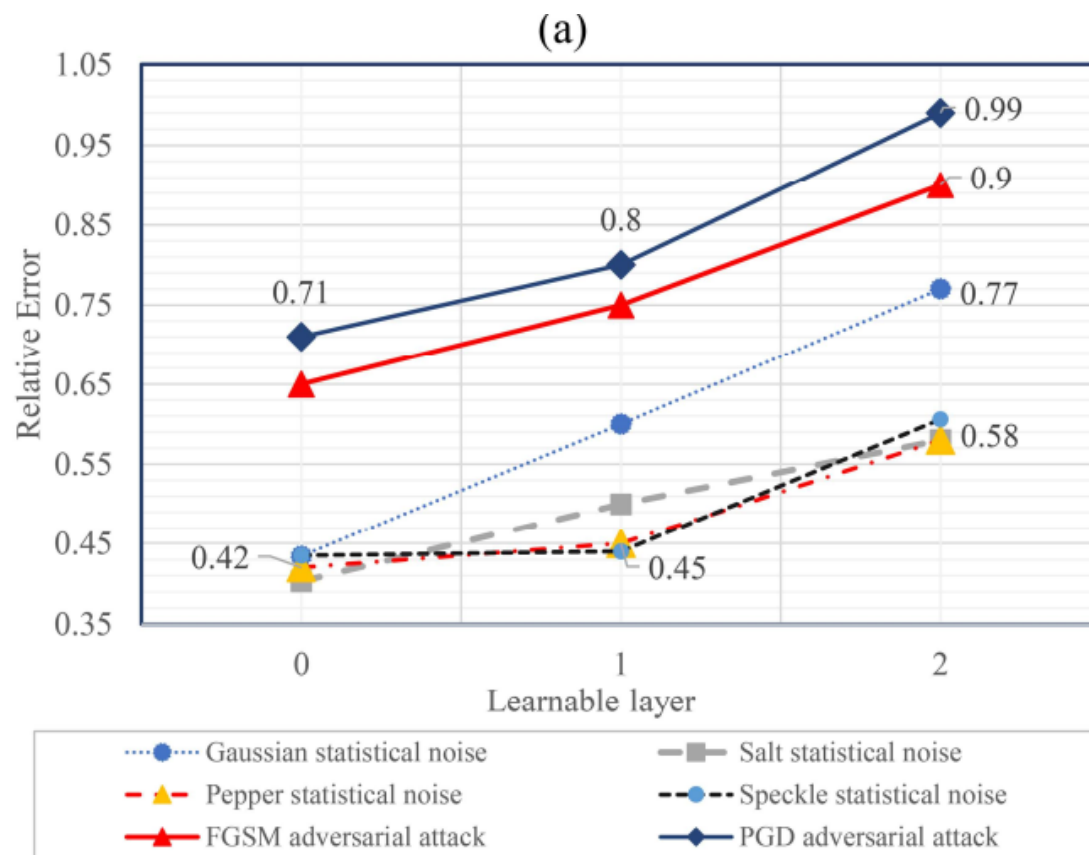
- Experiment architectures

**Table 1** Experiment architectures A, B, C, and D

Model A	$linear(100) \Rightarrow ELU \Rightarrow Linear(100) \Rightarrow ELU \Rightarrow Linear(100) \Rightarrow ELU \Rightarrow Linear(1)$
Model B	$Conv2D(16, (5 \times 5)) \Rightarrow ReLU() \Rightarrow Conv2D(32, (5 \times 5)) \Rightarrow ReLU() \Rightarrow MaxPool2D(2,2) \Rightarrow Conv2D(64, (5 \times 5)) \Rightarrow ReLU() \Rightarrow MaxPool2D(2,2) \Rightarrow Linear(100) \Rightarrow ReLU() \Rightarrow Linear(10)$
Model C	VGG-19 architecture <a href="#">[63]</a>
Model D	WideResNet <a href="#">[68]</a>



- Moon Dataset // model A (MLP)





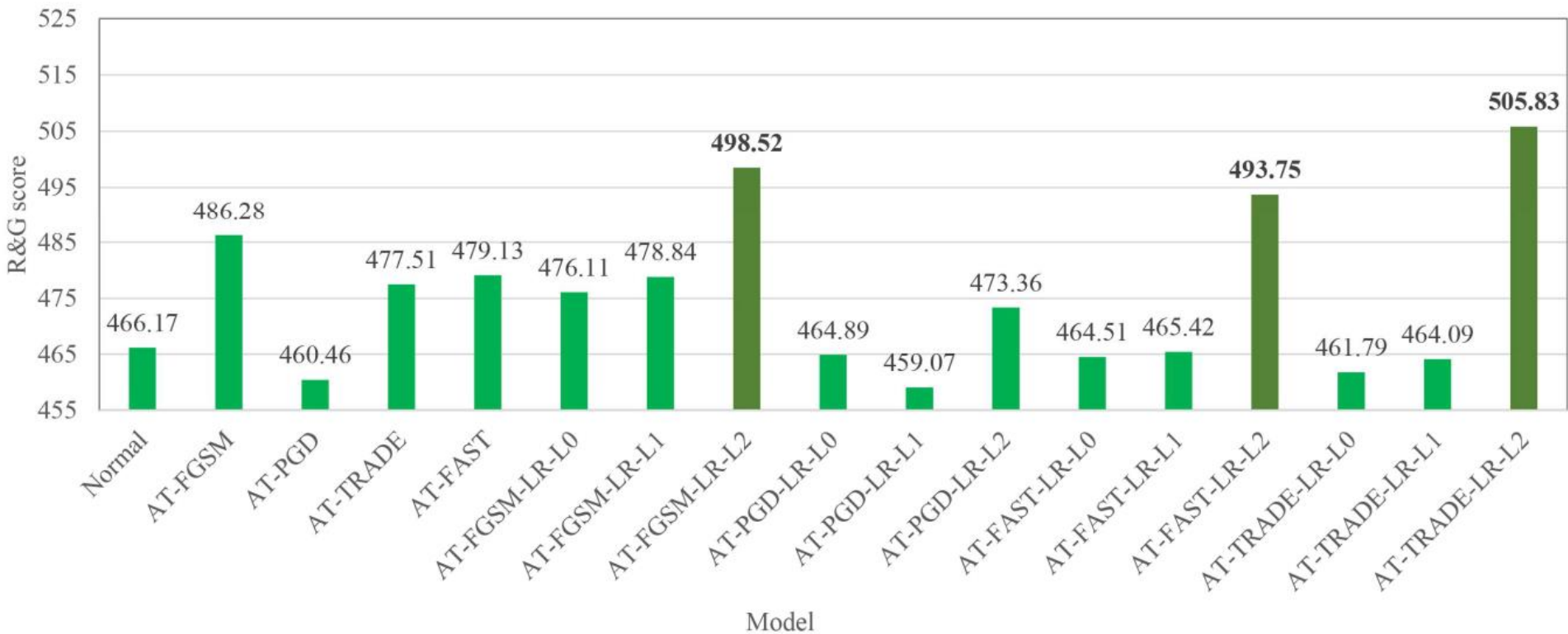
- Moon Dataset // model A (MLP)

**Table 2.** Evaluation of model *A* on Moon dataset with different loss functions against FGSM adversarial attack.

Training type	Accuracy of model A against FGSM with different epsilon values						R&G Score
	0	0.1	0.2	0.3	0.4	0.5	
Normal	<b>97.07</b>	<b>93.63</b>	82.5	76.83	63.79	52.35	466.17
AT-FGSM	<b>95.86</b>	<b>91.98</b>	<b>87.88</b>	<b>81.33</b>	70.54	58.69	<b>486.28</b>
AT-PGD	86.5	83.89	80.87	76.61	<b>72.57</b>	60.02	460.46
AT-TRADE	<b>96.17</b>	<b>93.06</b>	85.78	74.2	68.62	59.68	477.51
AT-FAST	<b>94.32</b>	89.72	84.91	78.35	70.18	61.65	<b>479.13</b>
AT-FGSM-LR-L0	92.81	90.25	85.36	80.74	68.21	58.74	476.11
AT-FGSM-LR-L1	91.87	89.77	<b>86.95</b>	<b>81.51</b>	69.15	59.59	478.84
AT-FGSM-LR-L2	93.98	<b>93.01</b>	<b>89.12</b>	<b>84.01</b>	<b>75.15</b>	63.25	<b>498.52</b>
AT-PGD-LR-L0	87.02	84.02	80.64	76.22	71.12	<b>65.87</b>	464.89
AT-PGD-LR-L1	86.2	82.94	79.23	75.16	70.5	65.04	459.07
AT-PGD-LR-L2	88.78	86.93	81.25	77.01	<b>72.81</b>	<b>66.58</b>	473.36
AT-FAST-LR-L0	86.86	83.85	80.33	76.34	71.34	<b>65.79</b>	464.51
AT-FAST-LR-L1	87.91	85.43	82.44	77.38	70.01	62.25	465.42
AT-FAST-LR-L2	92.91	90.31	<b>86.26</b>	<b>81.53</b>	<b>75.09</b>	<b>67.65</b>	<b>493.75</b>
AT-TRADE-LR-L0	86.84	82.38	80.55	76.37	71.58	64.07	461.79
AT-TRADE-LR-L1	87.25	84.22	81.25	76.98	71.14	63.25	464.09
AT-TRADE-LR-L2	<b>96.67</b>	<b>93.2</b>	<b>86.56</b>	<b>81.25</b>	<b>79.5</b>	<b>68.65</b>	<b>505.83</b>



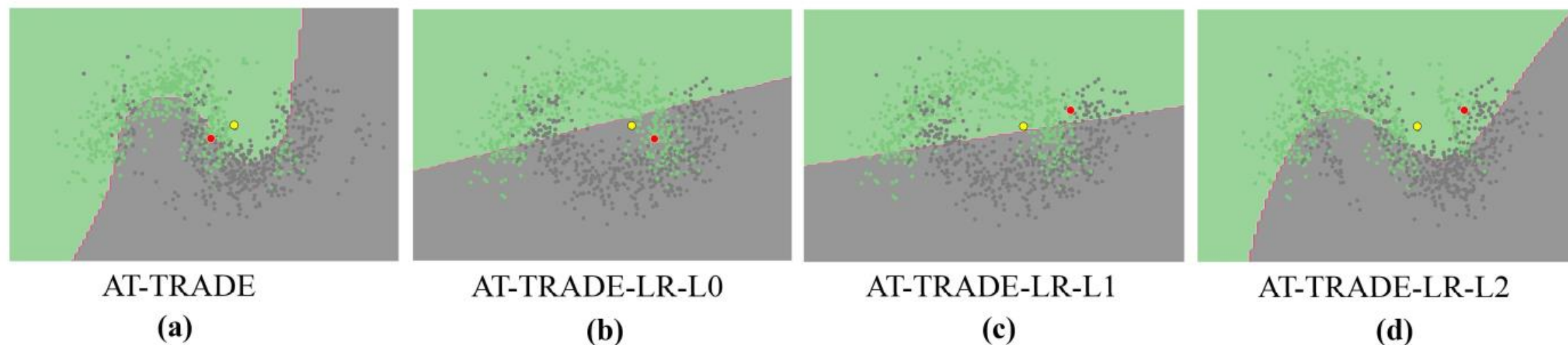
- Moon Dataset // model A (MLP)



**Fig. 6.** Comparison of R&G score results for different loss functions of architecture A on Moon dataset with different training loss functions.



- Moon Dataset // model A (MLP)

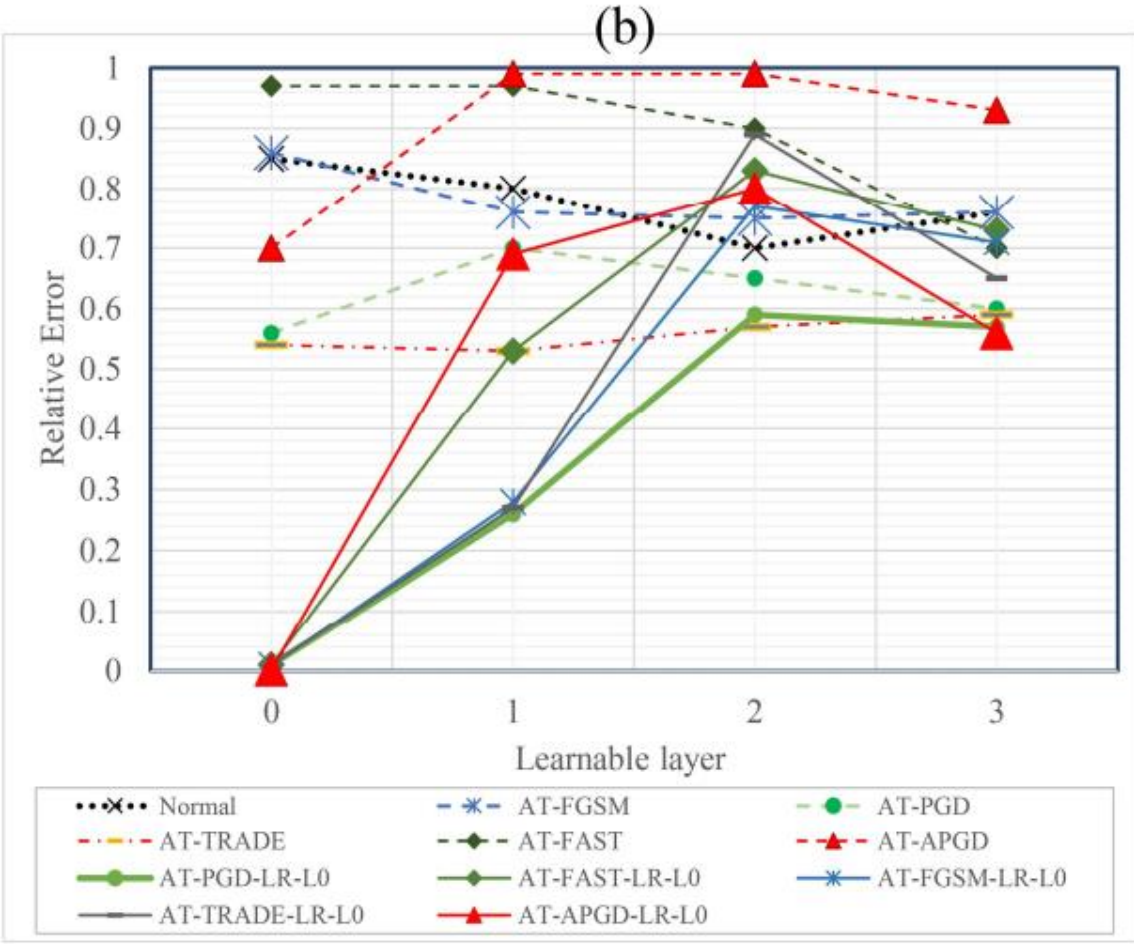
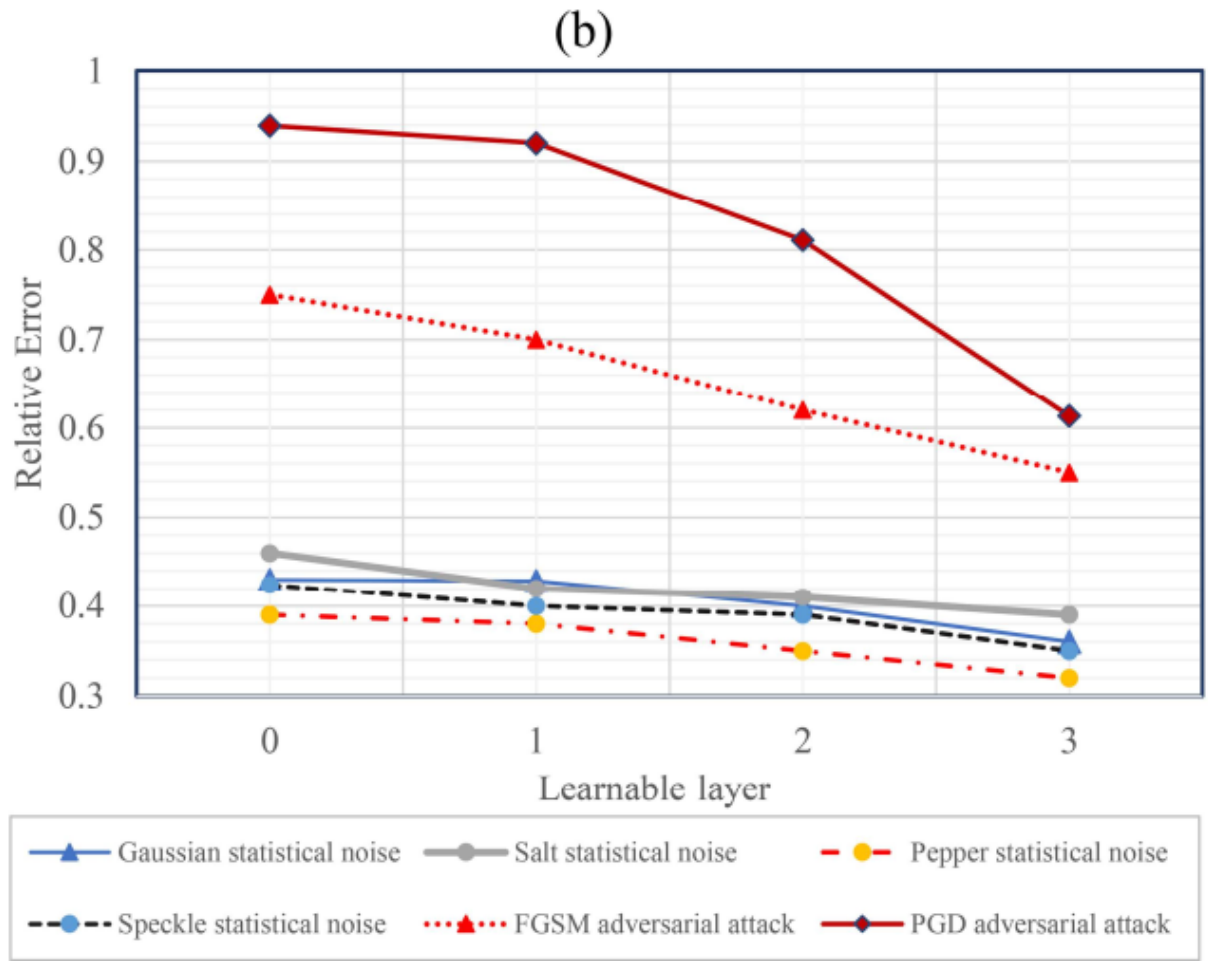


**Fig. 7.** Decision boundary of adversarial training with different loss functions on Model A.





- MNIST Dataset // model B (CNN)





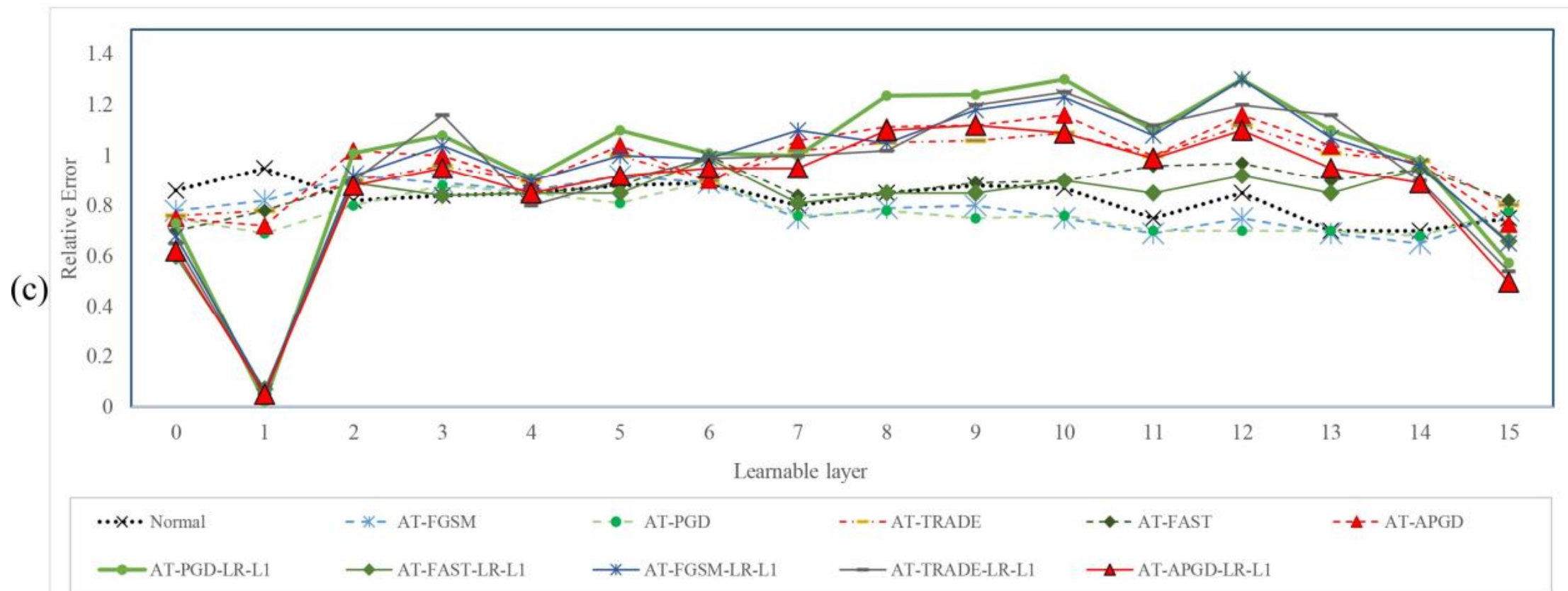
- MNIST Dataset // model B (CNN)

**Table 3.** Evaluation of model *B* on MNIST dataset with different loss functions against FGSM adversarial attack.

Training type	Accuracy of model A against FGSM with different epsilon values						R&G Score
	0	0.1	0.2	0.3	0.4	0.5	
Normal	98.82	82.1	47.2	17.87	6.96	4.25	257.2
AT-APGD	<b>99.36</b>	<b>98.54</b>	<b>97.67</b>	96.97	71.72	33.73	497.99
AT-APGD-LR-L0	98.97	98.11	97.36	<b>97.17</b>	<b>94.6</b>	<b>55.52</b>	<b>541.73</b>



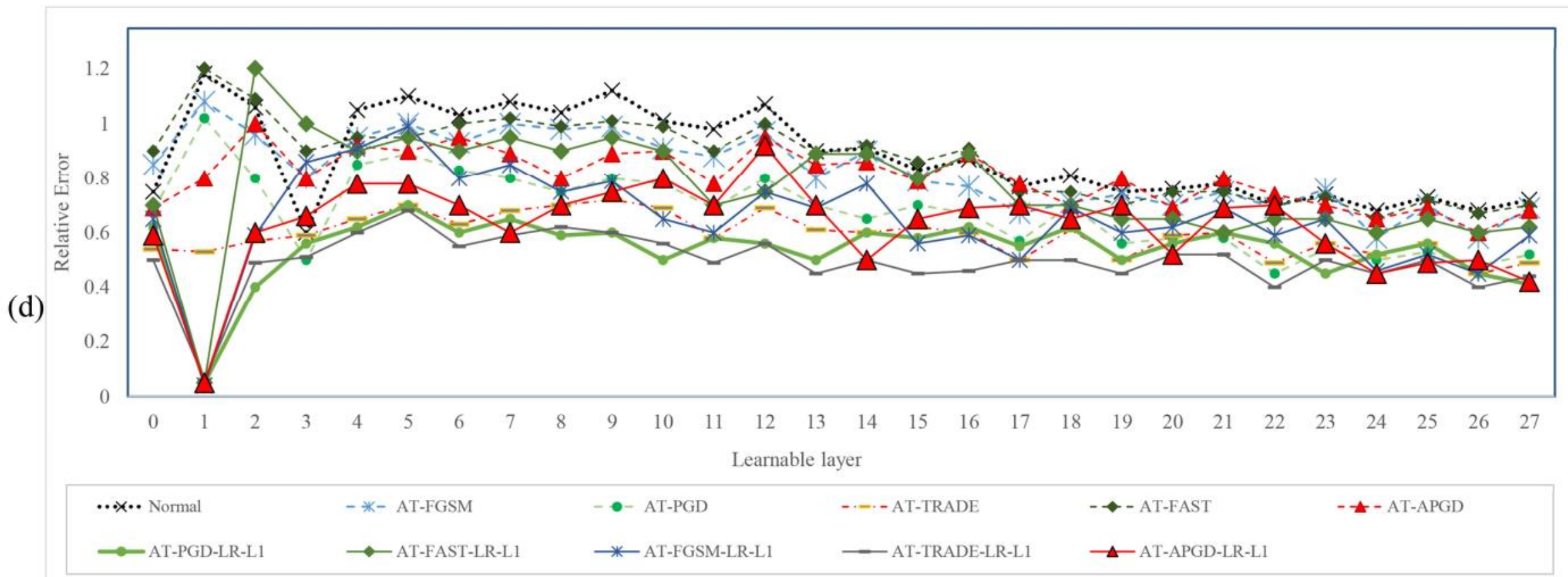
- CIFAR10 Dataset // model C (VGG19)





# Latest work: Experiment results

- CIFAR10 Dataset // model D (WideResnet)





- CIFAR10 Dataset // model D (WideResnet)

**Table 4.** Evaluation of models *C* and *D* on CIFAR-10 with different loss functions against FGSM adversarial attack.

Architecture	Training type	Accuracy of model C and D against FGSM with different epsilon values					
		0	0.01	0.03	0.1	0.2	R&G Score
Model C	Normal	<b>90.53</b>	48.91	43.91	31.5	22.29	237.14
	AT-APGD	83.68	74.69	72.21	50.04	41.45	322.07
	AT-APGD-LR-L1	83.55	<b>75.32</b>	<b>73.35</b>	<b>52.36</b>	<b>46.01</b>	<b>330.59</b>
Model D	Normal	<b>90.01</b>	40.01	36.35	22.3	16.2	204.87
	AT-APGD	82.60	50.12	<b>47.11</b>	41.46	40.01	261.3
	AT-APGD-LR-L1	81.81	<b>53.21</b>	48.17	<b>43.01</b>	<b>42.65</b>	<b>268.85</b>





**Mohammad Khalooei**

Mkhalooei [at] gmail.com

khalooei [at] aut.ac.ir

<https://ceit.aut.ac.ir/~khalooei>