# Enhancing Adversarial Example Transferability with an Intermediate Level Attack

Dependable AI Take-Home Minor-1

Kwanit Gupta, B19EE046[a]

[a]*Indian Institute of Technology, Jodhpur*

February, 2023

**Abstract**

Nowadays, adversarial attacks are of 3 types: white box, grey box, and black box on the basis of implementation details. Amongst these, the black box attacks are pretty hard to produce the results since they are heavily approximation-driven and their transferability sometimes becomes a question itself. Because of differences in neural architectures, their gradient-based behaviors on the feature maps differ uncontrollably. To facilitate this more on the universal ground, the respective paper's authors looked at the source model's intermediate layer upon which the baseline attack can be pushed into the target model's intermediate layer.

## 1. Introductory Guidelines

Dedicated submission guidelines are made for the same, where the following would be required in zip format:-

- Working codes, Readme
- Report, Presentation

### *1.1. Working Codes*

I referred to official ICCV 2019's GitHub implementation (Enhancing Adversarial Example Transferability with an Intermediate Level Attack). In addition to that, I made some convenient changes by cloning those codes into my profile (Modified Code-Base). In addition to that, the code base comprised of the following libraries/tech stacks:-

- Pytorch, Numpy, Pandas
- Tqdm, Argparse
- Skimage, Imageio, PIL

### *1.2. Readme Instructions*

As I ran the scripts of those codes within Google Colab's Ipynb Notebook, I mentioned the instructions and the dedicated notions such that step-by-step following those will lead to the dedicated results and graphs. Note that to save the CSV files, erase the "!rm test.csv" commands ahead of the main code snippets in each section.

## 2. Report Analysis

The main emphasis of ILA (Intermediate Level Attack) is to look upon the specific influential feature layer of the source model with the support of baseline attack, then utilize the notion towards the target models/layers (same or different). The authors have looked at the following baseline attacks:-

- IFGSM, Momentum IFGSM, and FGSM
- Carlini and Wagner
- Transferable Adversarial Perturbation [TAP]

They studied the above direction for the following models:-

- Alexnet, GoogLeNet, SENet18
- ResNet18, DenseNet121
- SqueezeNet 1.0
- Inception Networks

They choose CIFAR10 and Imagenet as benchmarks for ablation studies and experimentations (emphasized more towards CIFAR10).

Aligning with the interests of the course instructors and not deviating from the authors, I reproduce the results and implemented the scripts for the following benchmarks:-

- CIFAR10 (with the provided weights in .pth format)
- Tiny Imagenette-200 (Although a subset of Imagenet, but constitutes different inductive biases and sample-wise distribution)

### *2.1. Baseline Attacks*

ILA is indeed an approach or direction in which the control of adversarial attacks can be governed upon any specific neural network layer, without investing resources in the whole architecture. But for that to happen, it required the support of the following adversarial attack variants:-

- FGSM and its variants:- It utilizes the flawed assumption of a neural network possessing the high-dimensional linearity (i.e sum($\mathrm{w_i}$,$\mathrm{x_i}$)). Due to this, it generates adversarial perturbations by noticing the suitable direction of maximum gradient change for the model and being additive, it can easily be incorporated into the training input.

$$\delta = \epsilon sign(\nabla_x J_\theta(\theta, x, y)) \tag{1}$$

  Here, $\delta$ represents the perturbation, sign() as the signature function (+ or -), $\mathrm{J}_\theta$ as loss function associated with model parameters $\theta$, $\in$ as the bound set for perturbations, $\nabla_\mathrm{x}$ as the gradient function, x and y as input image and output label respectively.

- Transferable Adversarial Perturbations:- The TAP attack attempts to maximize the norm between the original image x and the adversarial example xat all layers. In contrast to our approach, they do not attempt to take advantage of a specific layer's feature representations, instead choosing to maximize the norm of the difference across all layers.

- Carlini and Wagner:- Nowadays, defensive distillation are used for the Complex Models to transfer their parameters and characteristics into a simpler dummy model, thereby preventing attacker to directly look on the original model.
  In order to breach it, this algorithm constrains $\mathrm{L_0}$, $\mathrm{L_2}$ and $\mathrm{L_\infty}$ norms.

$$\begin{aligned} Min.\ D(x, x+\delta) + c.f(x+\delta) \\ s.t.\ x+\delta \in [0,1]^n \end{aligned} \tag{2}$$

  Here, D represent the constraint paradigms or functions for $\mathrm{L_0}$ (helps to constraint the clean example points converted during the generation process), $\mathrm{L_2}$ (limits the overall degree of perturbations) and $\mathrm{L_\infty}$ (bounds the maximum permissible perturbation per pixel) norms, x and $\delta$ represent the input image and perturbation respectively, c denotes the configurable hyperparameter, f represents the set of varied objective functions.

### *2.2. Source and Target Models*

The authors carefully chose the architectural variations through which they showcased the impact of ILA, via the following characteristics:-

- Residual Connections in ResNets, SENets and DenseNets:- In many of the research papers related to these architectures, they indeed have the role of conveying original feature information to further layers (consecutively for ResNets and layer-by-layer for DenseNets). Since the adversarial attack impacted the original feature maps too (at the 1st layer), the effects of adversary will eventually pass on to future layers, due to which these networks are highly vulnerable to attacks.

- Vanilla Convs and Pools in Alexnet:- Being the fundamental neural network amidst others, it doesn't additionally introduce or retain the adversarial features. It indeed tries to differ and mitigate those by processing through convolutions and pooling layers.

- Parallel Conv Channels in Inception Networks and GoogLeNet:- For expressing different spatial mediums of information with respect to images and videos, these networks inculcate parallel convolution channels with their own sets of pooling combinations. But this can also lead to the introduction of their own artifacts while dealing with adverse feature maps, thereby ruining the characteristic keypoints.

- Fire Modules in SqueezeNets:- As the name suggests, it comprises of squeeze convolution filters (1x1) and expand convolution filters (1x1 and 3x3) that "ignite" the features or indeed amplify it. But this can also lead to amplified adversarial perturbations too.

### *2.3. Benchmarks*

A brief overview of the tested benchmarks is as follows:-

| $Dataset$ $(Data)$ | $Training$ $(Data_T)$ | $Testing$ $(Data_t)$ | $Classes$ $(N_c)$ |
|---|---|---|---|
| CIFAR10 | 50K | 10K | 10 |
| ImageNet | 1.28M | 100K | 1K |
| Imagenette 200 | 100K | 10K | 200 |

**Table 1:** Brief Overview about Benchmark Datasets

### 2.4. *Intermediate Level Attack*

To explain the working of ILA, the following is the pseudo-code or indeed an algorithm:-

- Algorithm requires the following information and parameters:-
  - Original Image and Adversarial Image obtained by baseline attack towards source model
  - Extractor Function to output the intermediate feature layer
  - N iterations, lr Learning Rate, and L-infinite bounded epsilon.
  - L Loss function (either ILAP or ILAF)
- Till the n number of iterations, calculate the vector difference formed by the pair of the Original Image - Adversarial Image of the Source Model and the pair of the Original Image - Adversarial Image of the Target Model.
- Change the Adversarial Image of the Target Model by the signed constant (sign obtained by the gradient of loss function L).
- Add the Original Image Matrix with the Clipped Difference of Perturbation (between the Adversarial Image of the Target Model and the Original Image).
- Clip the obtained matrix according to the range of image pixels (between 0 to 255).

To understand more about the Loss Function L, the following were the variations:-

#### 2.4.1. *ILAP (Intermediate Level Attack Projection Loss)*

The Vector difference obtained above would be required to mimic the same behavior, despite being different network architectures. To make that possible and simultaneously maximize the projection norm, the sign of the gradients are taken from the following Loss Function:-

$$Ł(f_1, f_2) = -f_1 . f_2 \tag{3}$$

Hereby, $f_1$ and $f_2$ represent the vector differences.

#### 2.4.2. *ILAF (Intermediate Level Attack Flexible Loss)*

Vector difference is optimally not sufficient to drive down the nature of perturbations on the target model. To improve it even further, fidelity of the perturbations at the layer l would be required to maintain alongside norm maximization via following Loss Function:-

$$Ł(f_1, f_2) = -\alpha . \frac{||f_1||}{||f_2||} + . \frac{f_1}{||f_1||} . \frac{f_2}{||f_2||} \tag{4}$$

Hereby, $f_1$ and $f_2$ represent the vector differences. Also, first term tries to maximize the synchronization between both distributions and second term is the projection of unit vector differences.

## 3. Results and Ablation Study

Most of the significance with analysis and insights are seen with CIFAR10, but the lesser idea was with tiny-ImageNette 200 because most of the accuracy metrics came out to be around 0% there (because of pipeline limitations or annotations file-data file mismatch). Authors indeed used layer indices, since they targetted blocks over convolution or pooling layers for simplified experimentations. Following would be the key observations:-

### 3.1. *FGSM variants on CIFAR10*

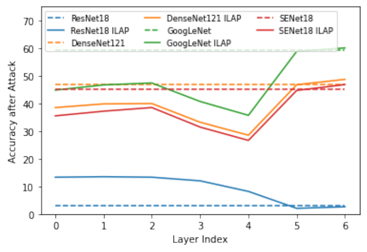
Overall Accuracy dropped around 4th Block for each target model (minima of plots were seen around layer index=4). If the source and target model architecture match each other, the minima observed were even more intensified. Also, as said earlier, the non-residual networks like GoogLeNet performed better than DenseNet and ResNet variants.

### 3.2. *FGSM variants on Tiny ImageNette 200*

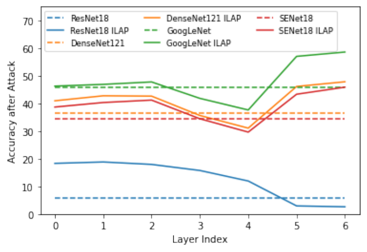
Overall Accuracy kept around 0% (despite taking lesser classes, but even after clipping the y-axis from 0 to 3, fluctuations weren't observed). But for some indices, the maxima presence was there (not more than 0.5%).

### 3.3. *Effect of Projections on TAP of CIFAR10*

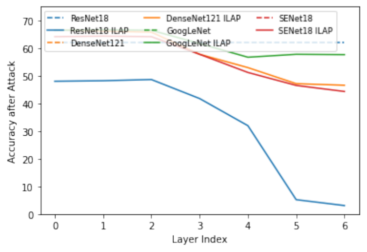
Projection, hereby meaning the introduction of a Flexible term in Loss (which maintained the fidelity of adversarial image produced with target model). It managed to penetrate even more than without fidelity term, and that too by 15-20% (observing the maxima of curves).



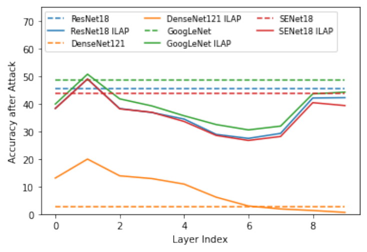
(a) IFGSM on Resnet18



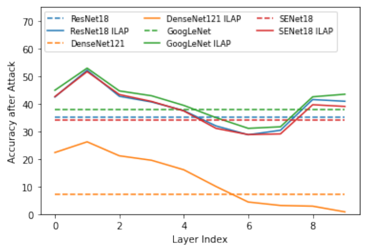
(b) Momentum IFGSM on Resnet18



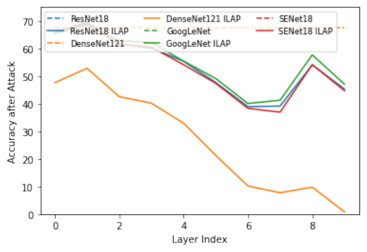
(c) FGSM on Resnet18



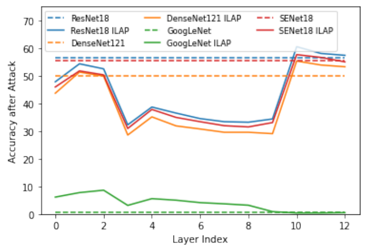
(d) IFGSM on DenseNet121



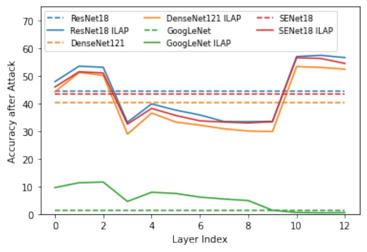
(e) Momentum IFGSM on DenseNet121



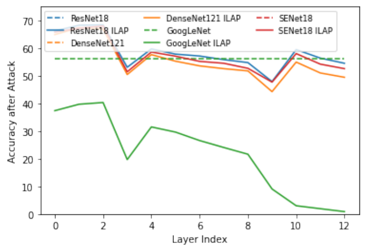
(f) FGSM on DenseNet121



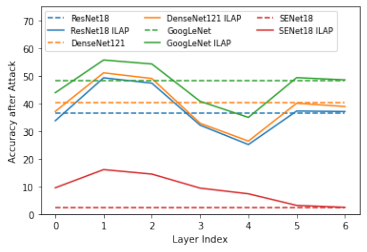
(g) IFGSM on GoogleLeNet



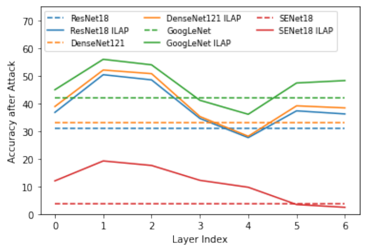
(h) Momentum IFGSM on GoogleLeNet



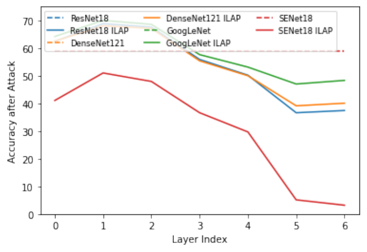
(i) FGSM on GoogleLeNet



(j) IFGSM on SENet18

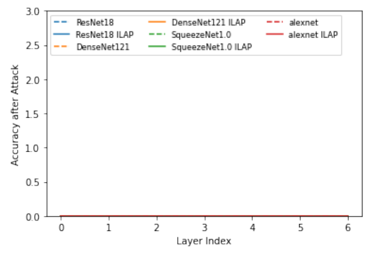


(k) Momentum IFGSM on SENet18

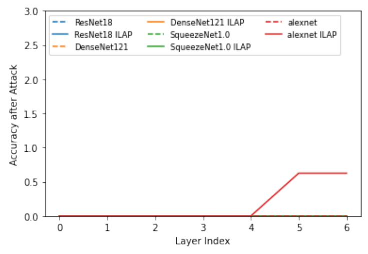


(l) FGSM on SENet18

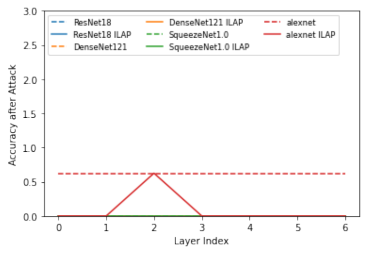
**Figure 1:** Intermediate Level Attack on CIFAR10 (each image symbolizing for mentioned source model on target models)



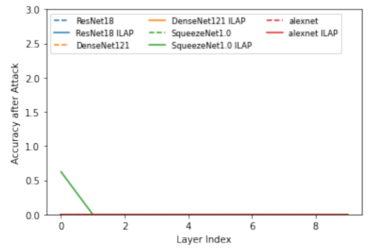
(a) IFGSM on Resnet18



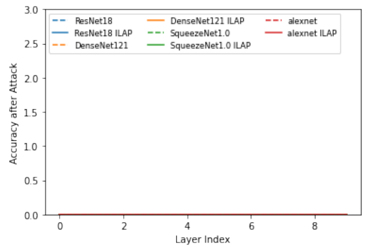
(b) Momentum IFGSM on Resnet18



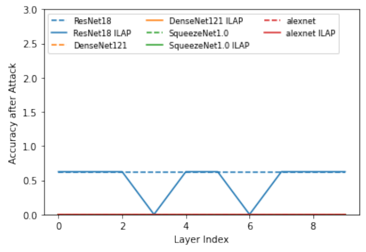
(c) TAP on Resnet18



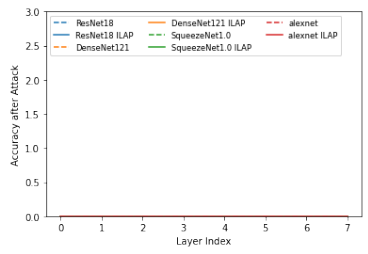
(d) IFGSM on DenseNet121



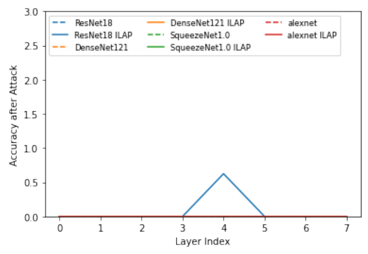
(e) Momentum IFGSM on DenseNet121



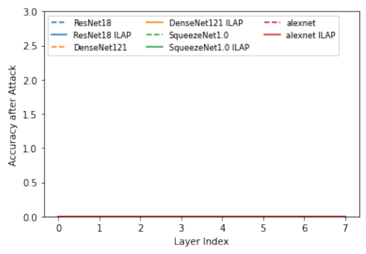
(f) TAP on DenseNet121



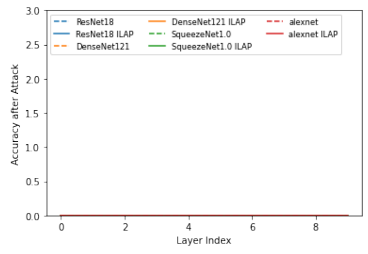
(g) IFGSM on Alexnet



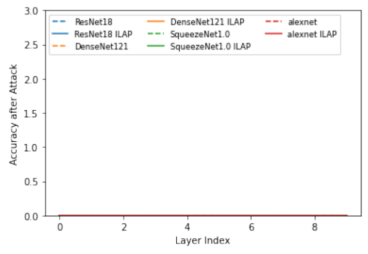
(h) Momentum IFGSM on Alexnet



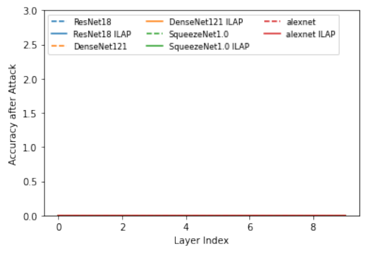
(i) TAP on Alexnet



(j) IFGSM on SqueezeNet 1.0

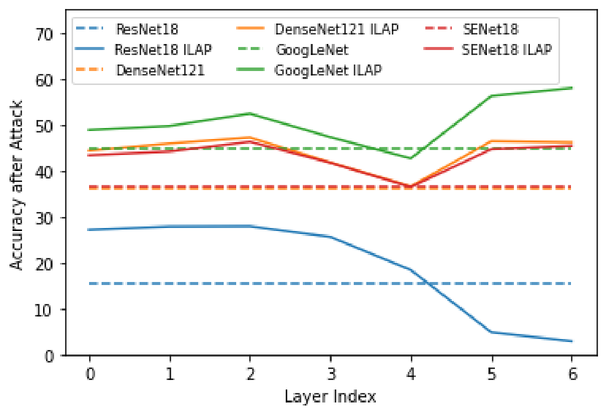


(k) Momentum IFGSM on SqueezeNet 1.0

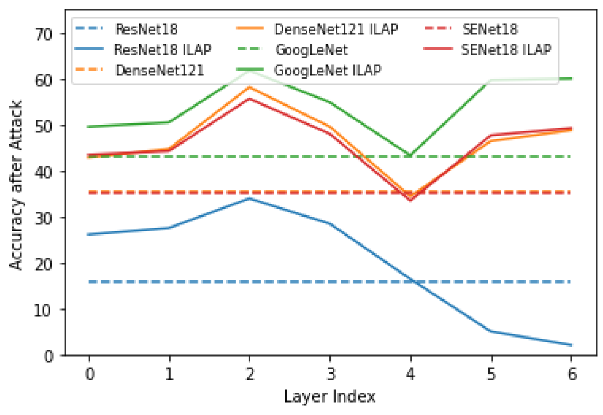


(l) TAP on SqueezeNet 1.0

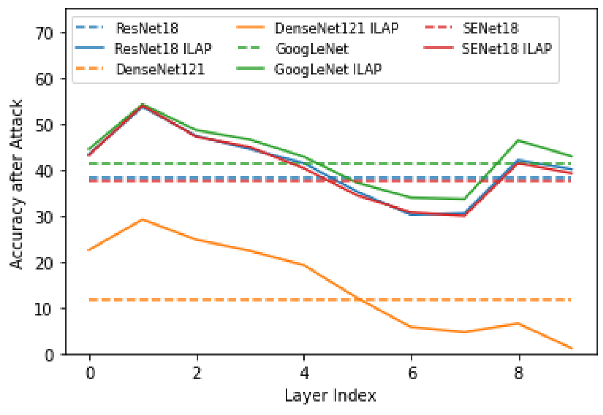
**Figure 2:** Intermediate Level Attack on Tiny-ImageNette 200 (each image symbolizing for mentioned source model on target models)



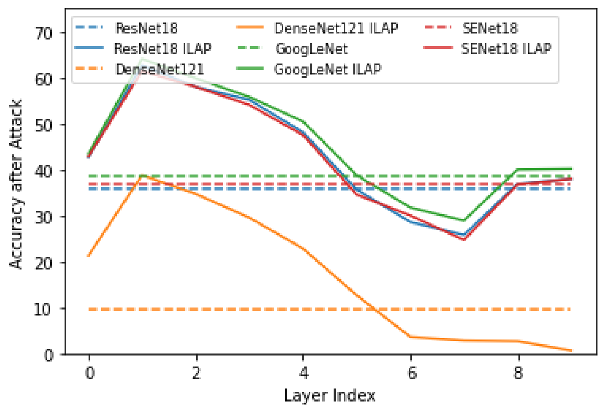
(a) TAP with Projection on Resnet18



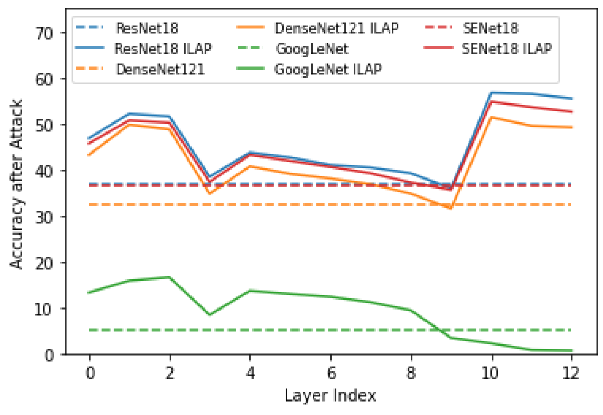
(b) TAP without Projection on Resnet18



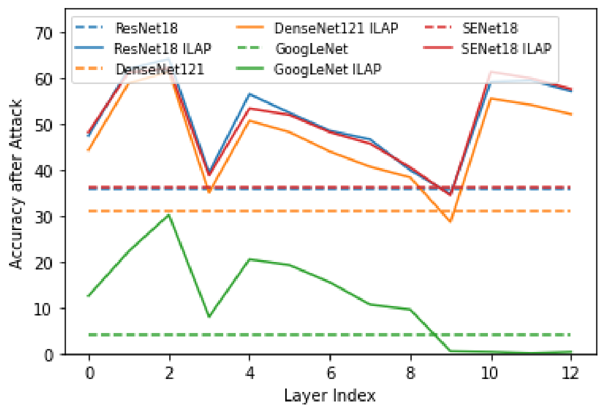
(c) TAP with Projection on DenseNet121



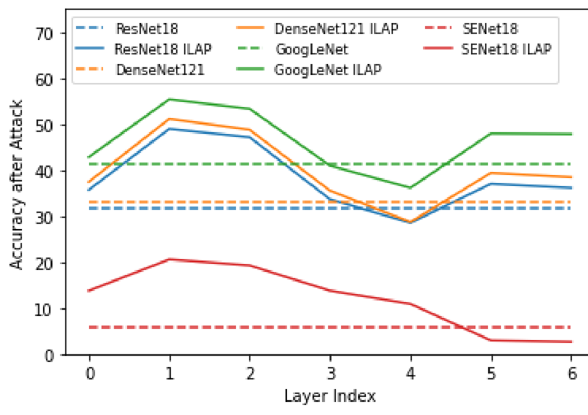
(d) TAP without Projection on DenseNet121



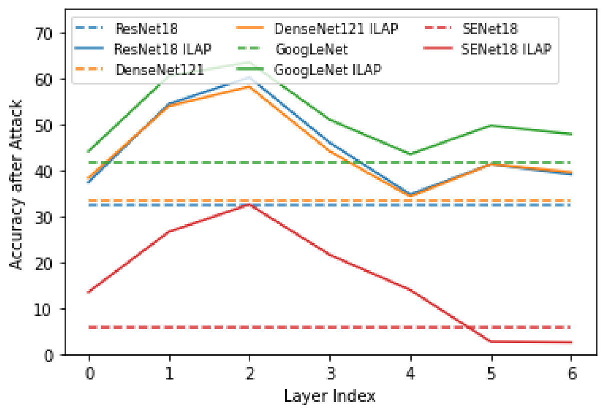
(e) TAP with Projection on GoogLeNet



(f) TAP without Projection on GoogLeNet



(g) TAP with Projection on SENet18



(h) TAP without Projection on SENet18

**Figure 3:** Effect of Projection on TAP for CIFAR10 (each image symbolizing for mentioned source model on target models)