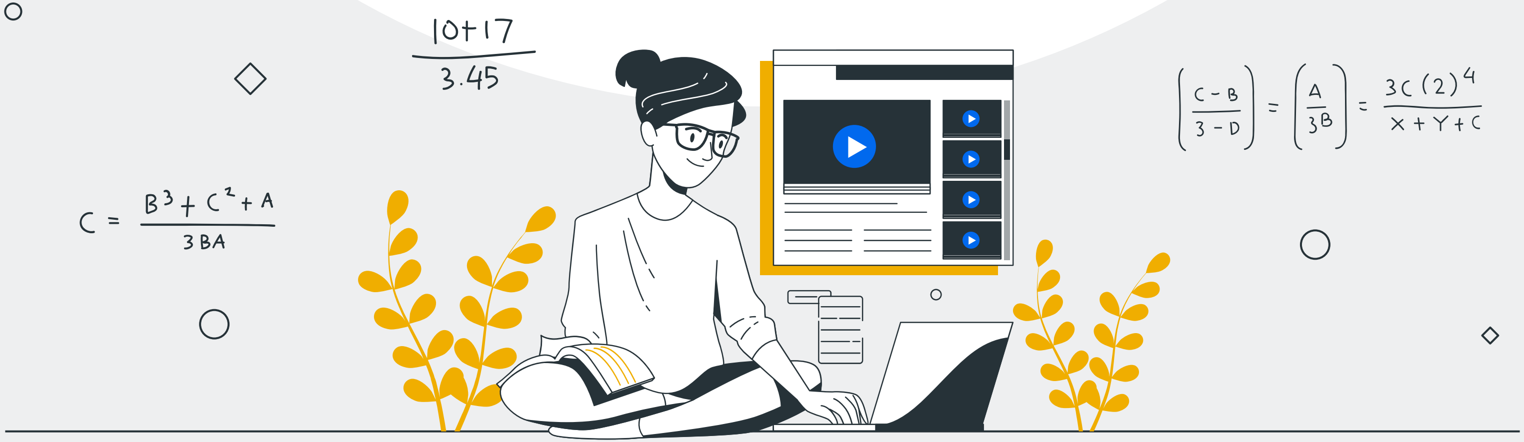
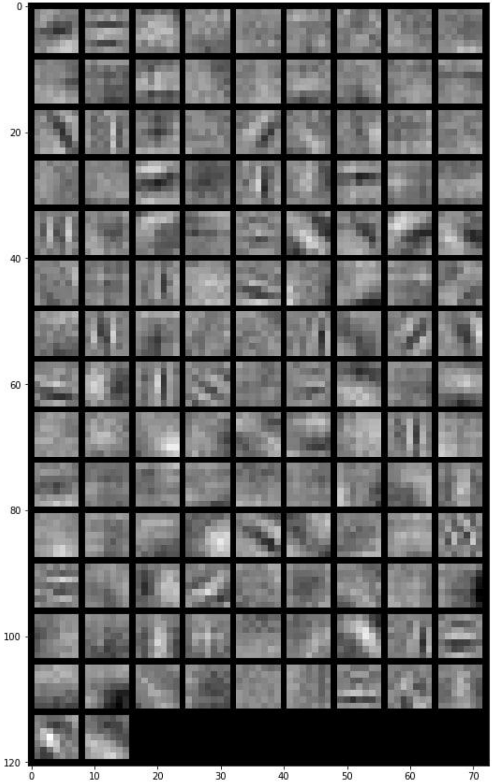
Adversarial Regularization for Convolution Filters



Motivation

Filters with patterns



CIFAR100

Kernel size: 7x7

Train size: 50 000

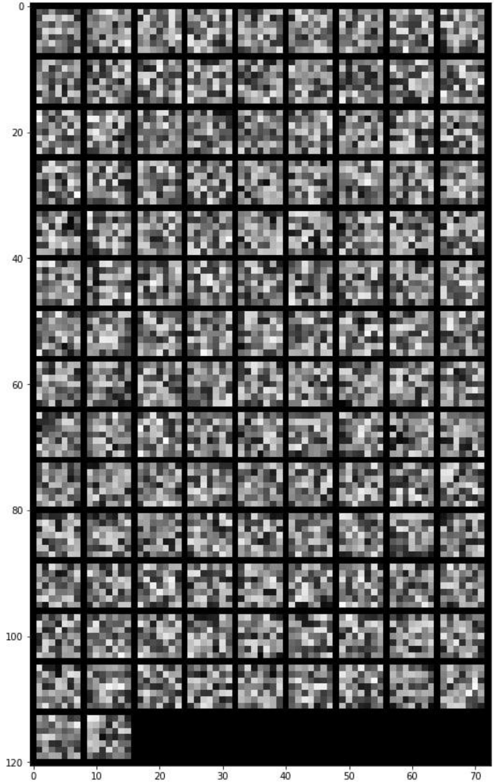
CNN

Filters without patterns

CIFAR10

Kernel size: 7x7

Train size: 100



Problem statement

Collect **good filters** and train a model on small data with **adversarial regularization** techniques by applying a **discriminator** to regularize **new filters** while training on **small data**.

01

Filters collecting

Collect good filters by training the CIFAR10 classification dataset with filters of size 7x7, 5x5 and random seeds of values 0 to 4.

02

Reduced dataset preparation

Prepare a new dataset with small training set based on CIFAR10 and retrain a classifier on a new reduced dataset.

03

Architecture development

Develop and implement an architecture of discriminator

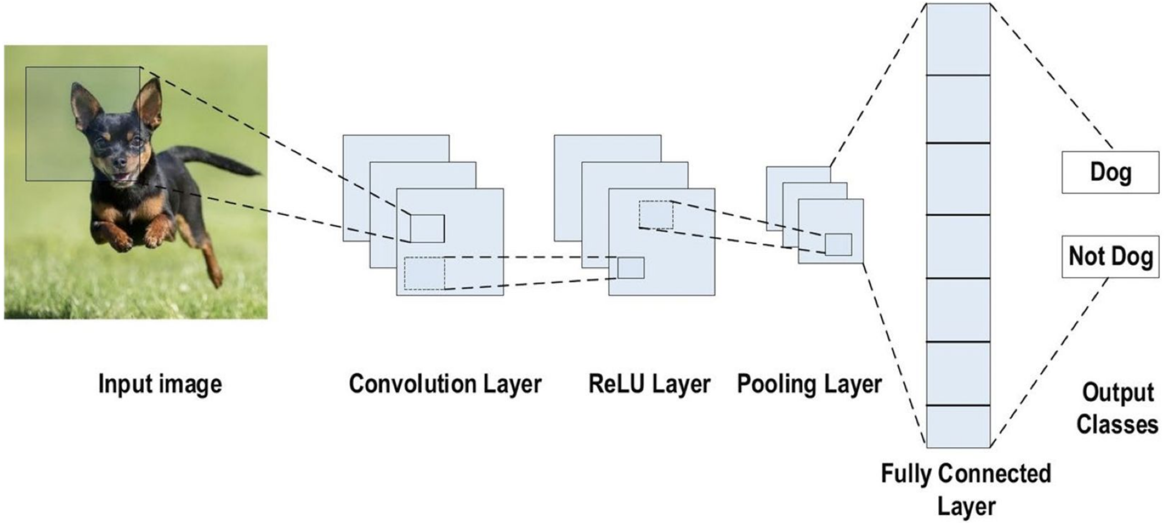
04

Full model training

Train a full model (classifier with discriminator for convolutional layer) on a reduced dataset.
Provide results of filter improvements.

Related work

Convolutional Neural Network (CNN) is a network similar to the **multi-layer perceptron**, which consists of numerous **convolution** layers preceding **pooling** layers, while the ending layers are **fully connected** layers.



Example of CNN architecture

[Alzubaidi, Laith, et al. "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions." Journal of big Data 8.1 (2021): 1-74.]

Key benefits of the CNN:

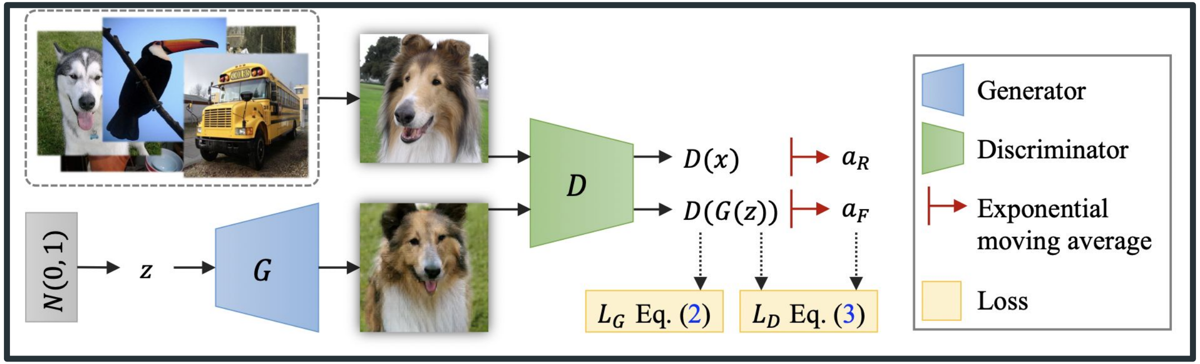
- Equivalent representations
- Sparse interactions
- Parameter sharing
- Automatically identifies the relevant features without any human supervision.

Reducing overfitting:

- Dropout
- Drop-Weights
- Data Augmentation
- Batch Normalization

Related work

Generative Adversarial Net (GAN) is an algorithm consists of two separated models **Generator** (G) and **Discriminator** (D), which have a structure of multilayer perceptrons



Example of GAN algorithm

[Tseng, Hung-Yu, et al. "Regularizing generative adversarial networks under limited data." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.]

Value function

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

GANs advantages:

- Markov chains are never needed
- Only backprop is used to obtain gradients
- No inference is needed during learning
- A wide variety of functions can be incorporated into the mode
- Can represent very sharp distributions

GANs disadvantages:

- G must not be trained too much without updating D

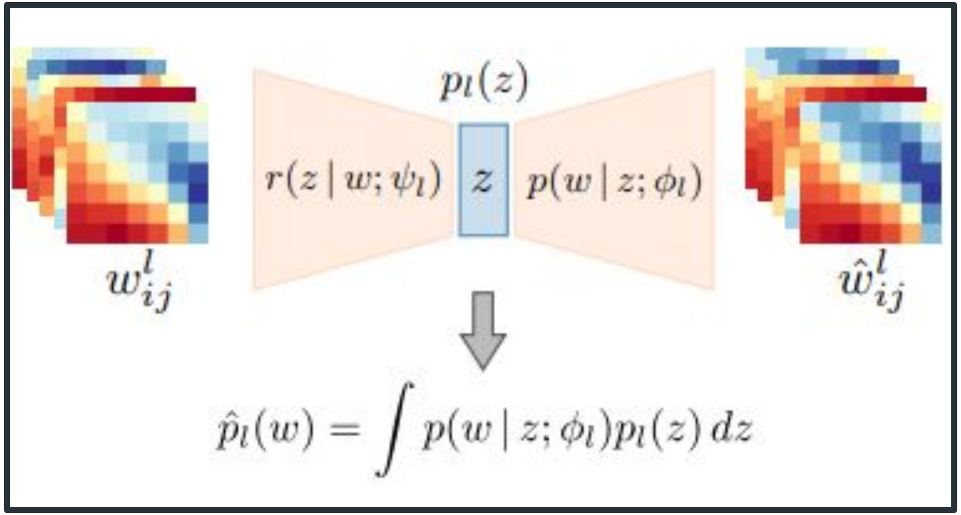
Training process:

G is trying to generate such a sample that the probability for D to make a mistake will be maximal.

[Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems 27 (2014).]

Related work

Bayesian inference is a tool which is used to transform a prior distribution over a model to posterior.



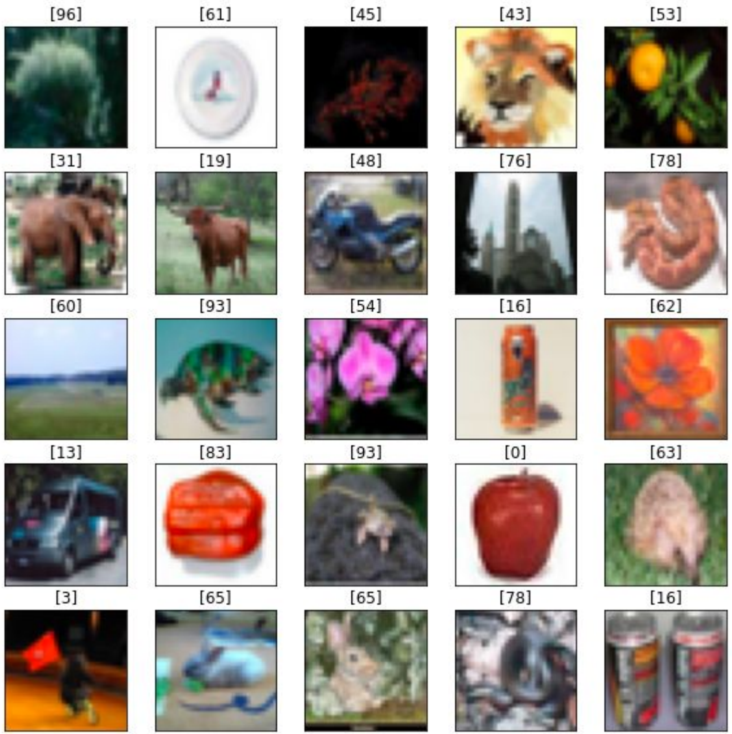
DEEP WEIGHT PRIOR:

- approximates the source kernel distribution
- incorporates prior knowledge about the structure of convolutional filters into the prior distribution

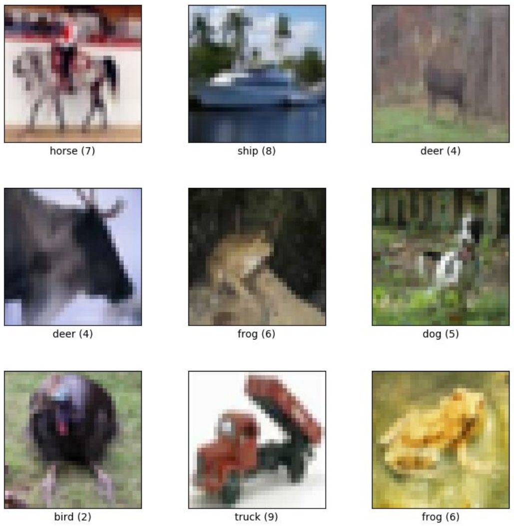
[Atanov, Andrei, et al. "The deep weight prior." arXiv preprint arXiv:1810.06943 (2018).]

- **Deep weight prior** significantly improves classification performance in the case of limited training data
- Initialization of conventional convolution networks with samples from a deep weight prior leads to **faster convergence** and **better feature extraction** without training i.e., using random weights.

Datasets descriptions

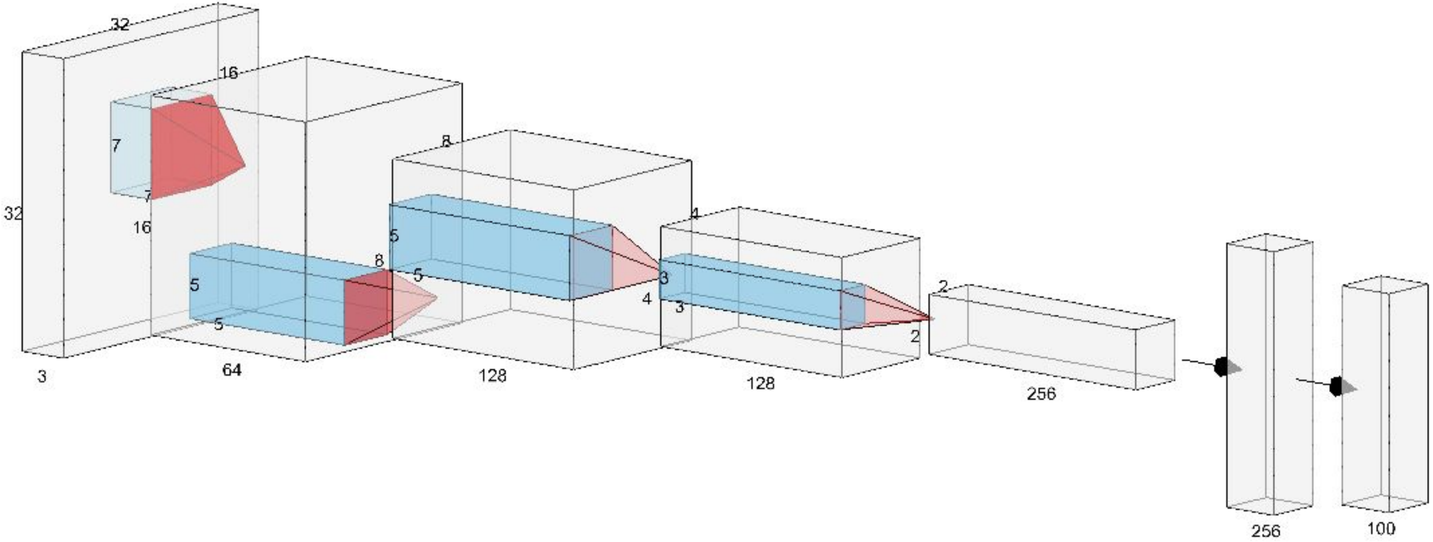


CIFAR-100



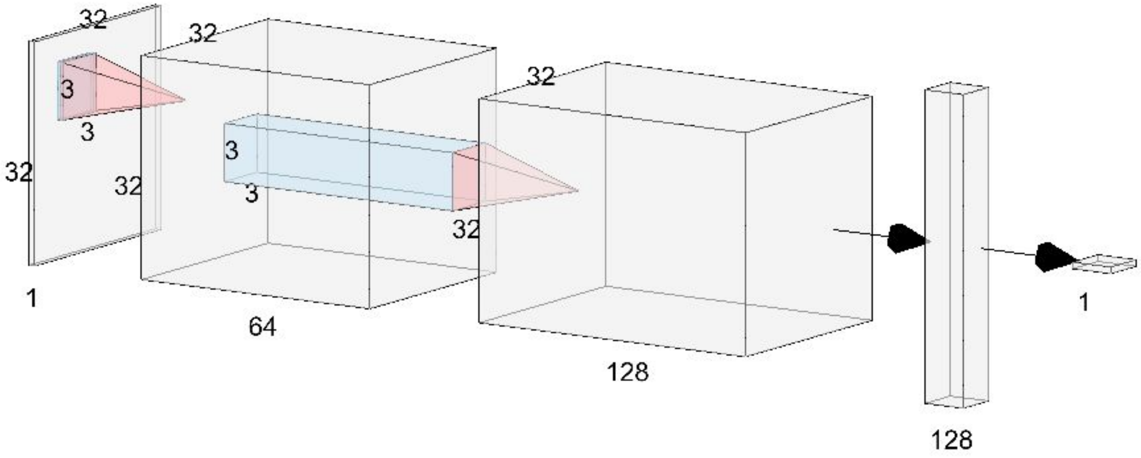
CIFAR-10

Models’ architecture



Classifier (Generator)

Models' architecture



Discriminator

Training with regularization

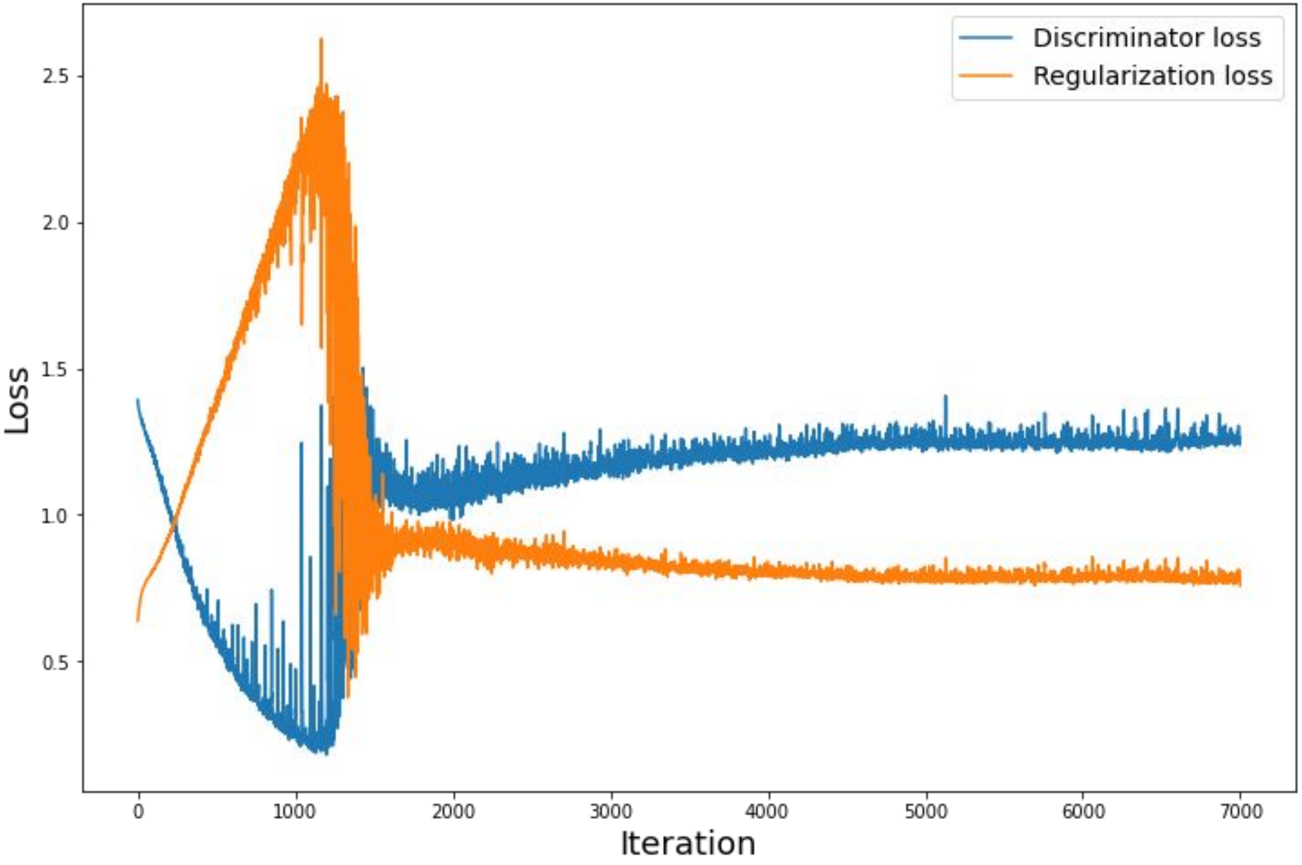
Algorithm 1 Epoch of training with regularization

for $imageBatch$ in $trainSet$ **do**
  $imagePred = classifier(imageBatch)$
  $loss =$ classifier loss on $(imageBatch, imagePred)$
  $realBatch =$ random batch of good kernels
  $fakeBatch =$ random batch of classifier kernels
  $loss+ = \lambda\cdot$ (generator loss on $fakeBatch$)
  Update weights of $classifier$ based on $loss$
  Train $discriminators$ on $realBatch$, $fakeBatch$
end for

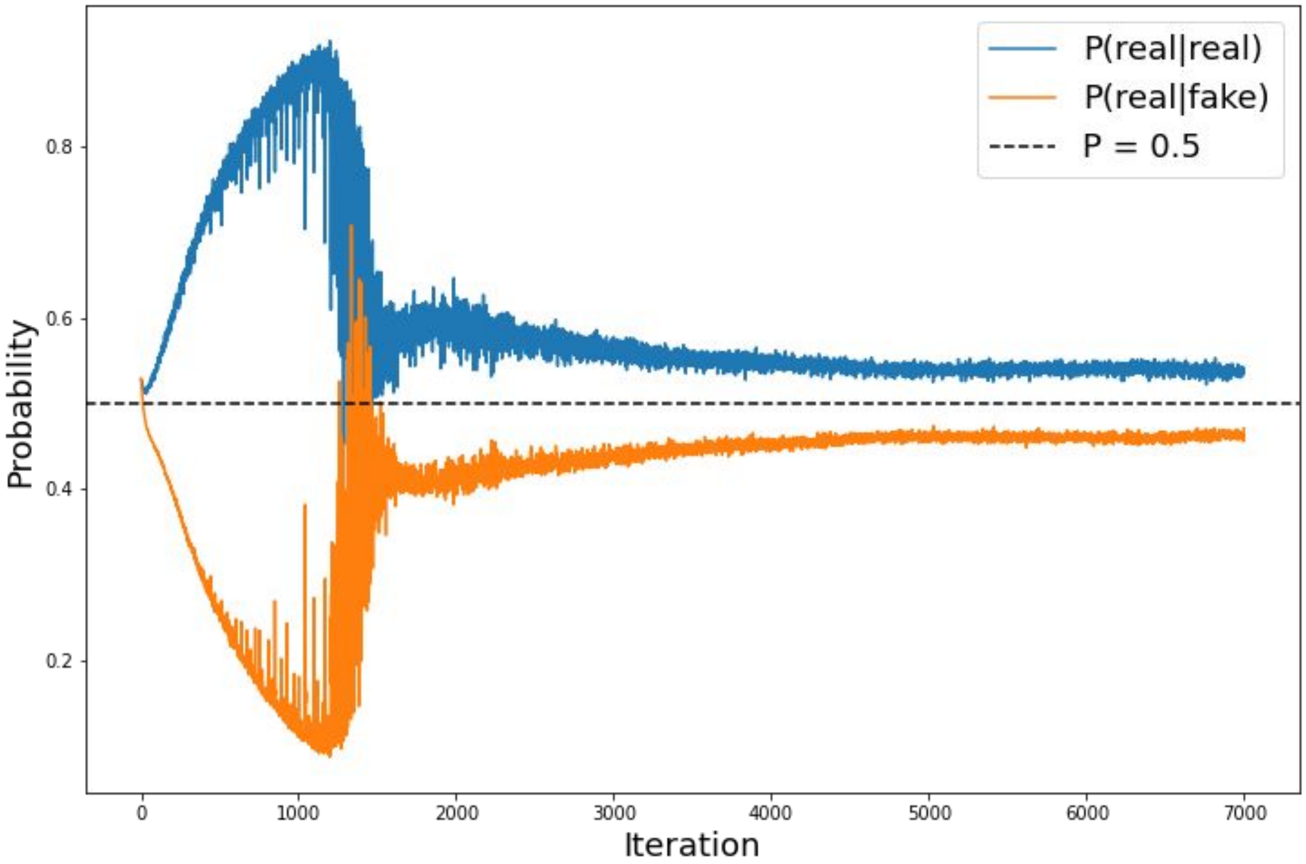
Experiments

1. Train 5 models on CIFAR-100 and collect kernels.
2. Train a model on 100 samples from CIFAR-10.
3. Train another model but with adversarial regularization
   One discriminator for the first convolutional layer (7x7)
   Regularization coefficient = 1

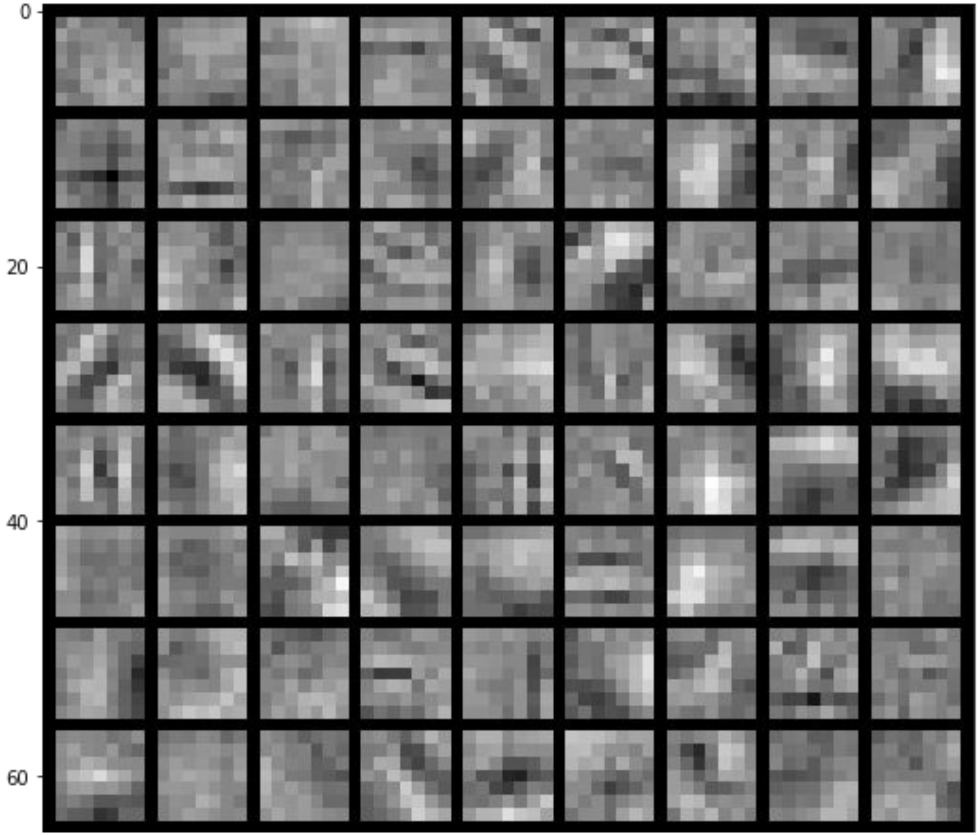
Experiments



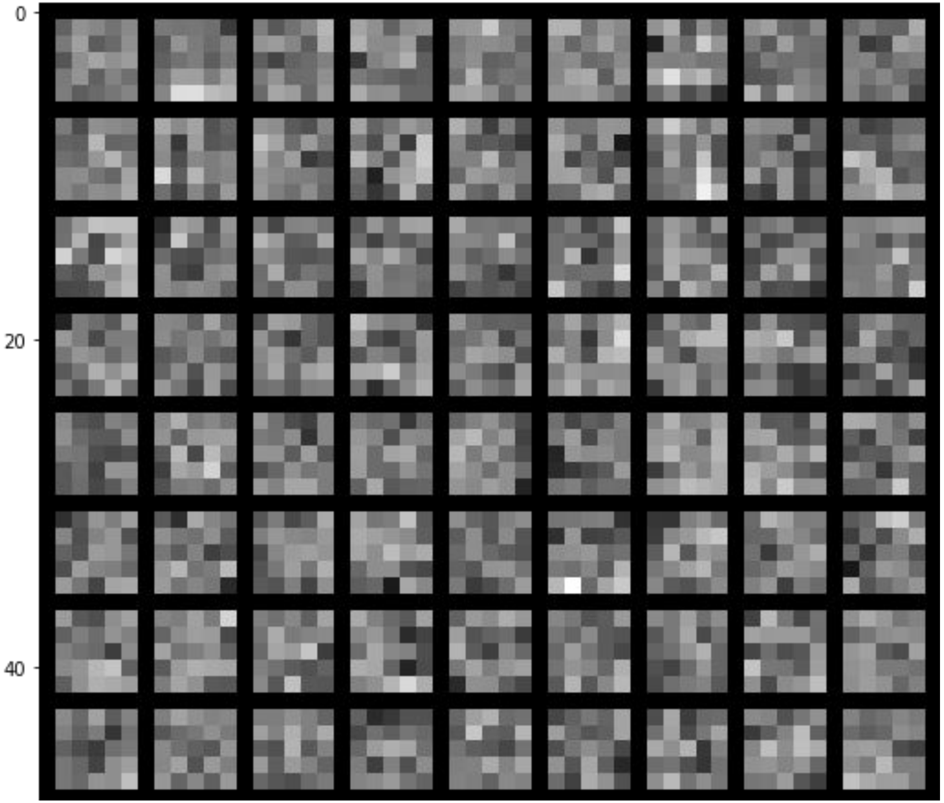
Experiments

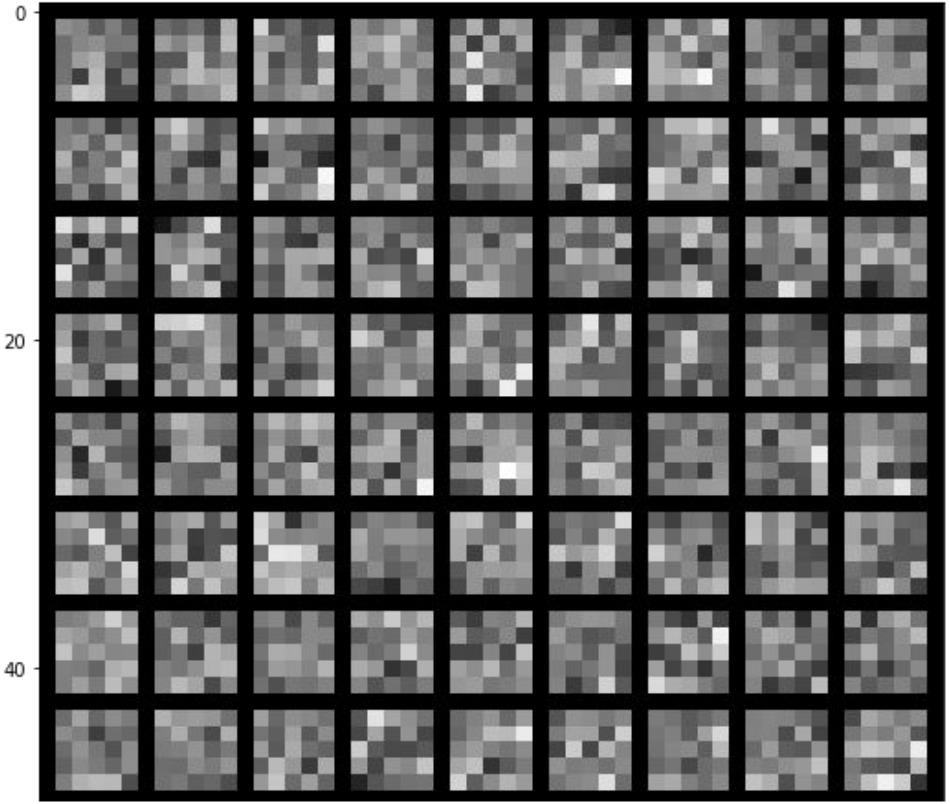


Obtained results

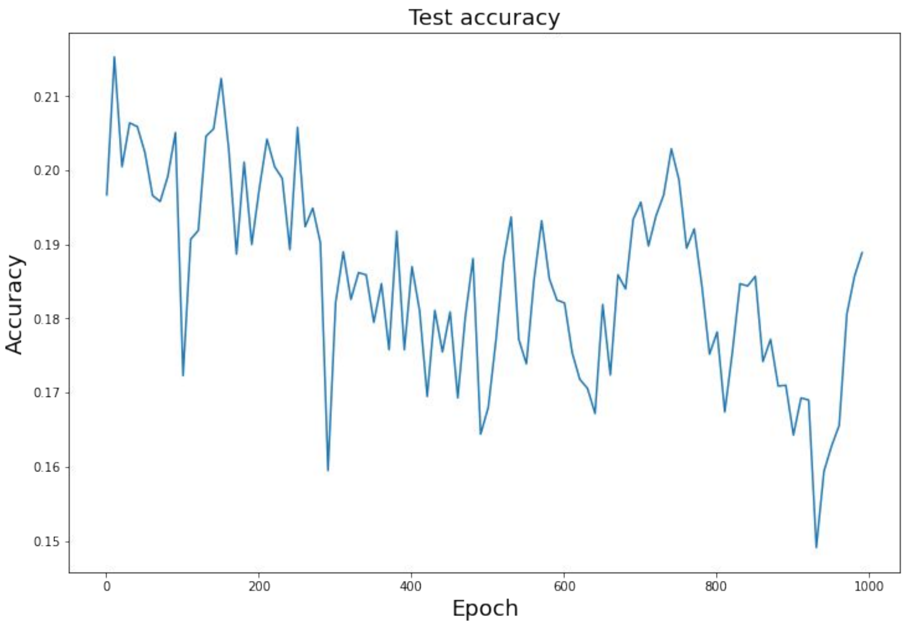


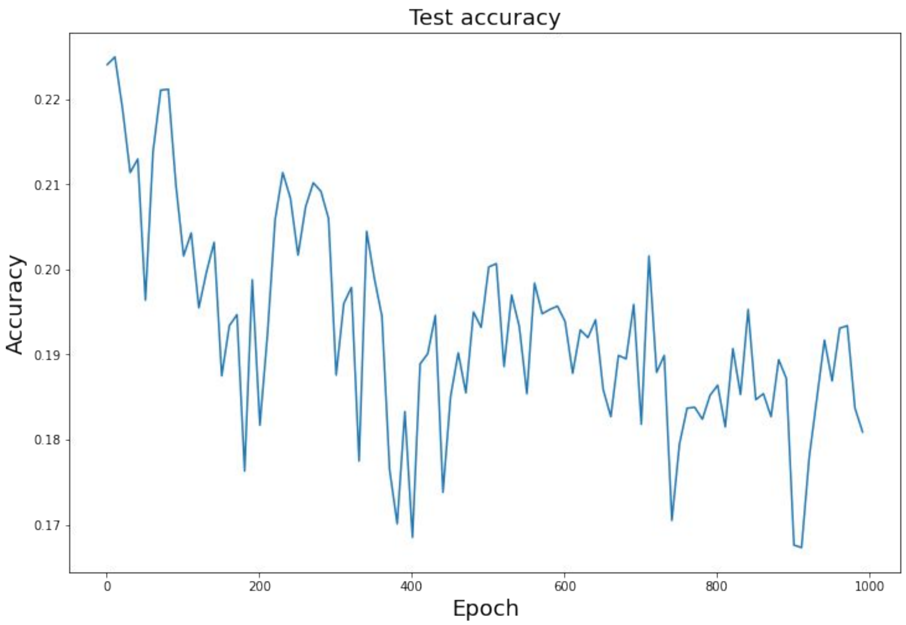
Obtained results





Obtained results





Thank you for your attention

Team 19

Dmitrii Gavrilev

Garsiya Evgeniy

Lina Bashaeva

Farid Davletshin

Dmitry Gilyov