DLE Homework 5
CNN Visualization and Adversarial Patterns

Vojtěch Michal, michavo3

Fig. 1: An original higher resolution image of a Labrator retriever.

Fig. 2: Resized and resampled image appropriate for image classification.

***Abstract*—The assigned homework[1] focuses on visualization of various aspects of Convolutional Neural Netowrks (CNNs) and search for Adversarial Patterns. All experiments are performed on the VGG11 convolutional neural network.**

I. Classification Baseline

The pretrained VGG11 model was used to classify the image of a Labrador retriever shown in Fig. 1. As analyzed in previous lab reports, preprocessing is an important step required for best performance of learning tasks, hence the image was resampled and normalized to the state shown in Fig. 2, showing slight change of color channel intensities. Feeding the tensor as an input to the model yielded class predictions and the corresponding confidences listed in Table I.

Prediction order	Class name	Class number	Confidence
1	Labrador retriever	208	70.02 %
2	golden retriever	207	11.26 %
3	Chesapeake Bay retriever	209	9.70 %
4	bloodhound, sleuthhound	163	0.78 %
5	Weimaraner	178	0.61 %
6	kelpie	227	0.59 %
7	kuvasz	222	0.55 %
8	tennis ball	852	0.51 %
9	Rhodesian ridgeback	159	0.44 %
10	redbone	168	0.42 %

TABLE I: Classification of the unmodified image in Fig. 1

II. Visualization of Model's Activation Channels

Visualizing the activations during a forward pass through the network may shed some light on the model's internals and visual features the model finds "interesting". Due to large dimensionality of activations (hidden convolutional layers have up to 512 channels), only the l_2 norm of all channels can be reasonably visualized. Such visualizations of all feature layers for the dog image tensor from Fig. 2 are shown in Fig. 3. The model is clearly sensitive mostly to the sharp object's outline, gradually blurring it into abstract, unrecognizable shapes. This is related to the receptive field – individual convolutional layers use kernel size 3x3 and max pool layers use kernel size 2x2. Therefore, individual visual features start very local (small receptive field of the first layers) and gradually become larger and less sharp. As a side note, the effect of the max pool layer is nicely visible from the last image (activation of layer 20), where a region of high-activation nodes is replaced by one big node.

To visualize the gradients, the tensor was propagated forward through the rest of the model, followed by backward pass from the calculated score corresponding to the true image class (Labrador retriever). Gradients are visualized in the Fig. 4.

[1]The homework assignment is available on https://cw.fel.cvut.cz/wiki/courses/bev033dle/labs/lab4_visualization/start

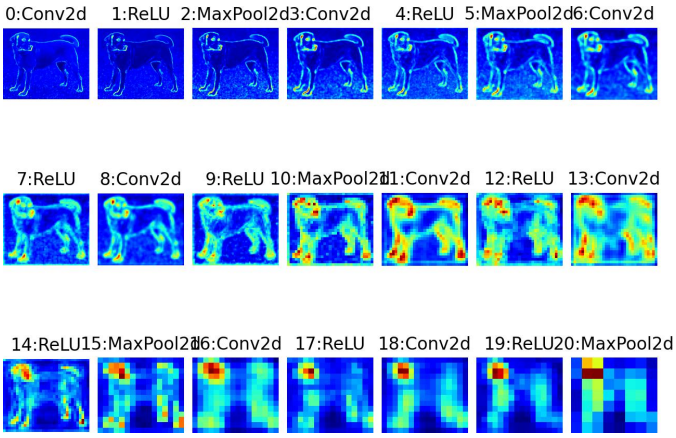


Fig. 3: Visualized feature maps of the model.

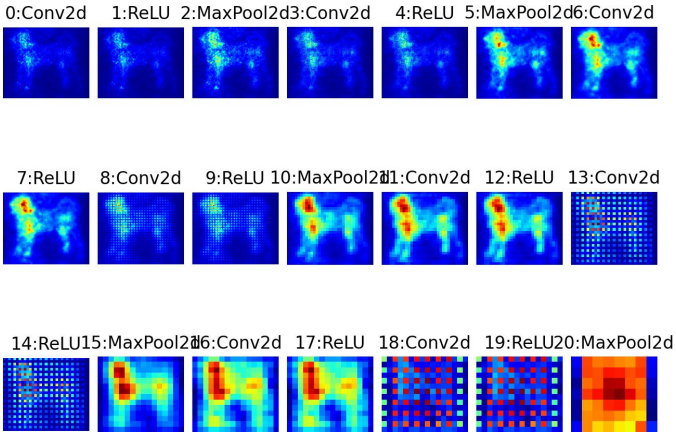


Fig. 4: Visualized gradients of the true class's score w.r.t. individual layer activations.

III. Patterns Maximizing Channel Activations

Visual patterns that maximize activations of several channels on the output of the 9-th hidden layer are shown in Fig. 6 and 7. Non-smooth patterns are less constrained than smooth patterns, hence they achieve comparably larger activations, as shown in Fig. 5.

IV. Adversarial Attack

The last section of this report describes a successful adversarial attack against the model. A clear image of the Labrador retriever (class number 208) shown in Fig. 8a was slightly distorted with noise invisible to human eye that would convince the model that the image in fact shows a wall clock (class number 892). The distorted image shown in Fig. 8b is truly indistinguishable from the clean image by the human eye. Their difference is visualized in Fig. 9 (normalized to the $[-1, 1]$ range in Fig 9a and left in absolute values in Fig. 9b). While the human eye is very bad in distinguishing similar high-frequency visual data, its presence can be fatal for the operation of the machine learning algorithm. The Table II contains parameters of the noise the attacking algorithm added to the clean image.

The Adam optimizer with a safe value of learning rate $l_r = 3 \cdot 10^{-4}$ was used for all optimization runs. The targeted

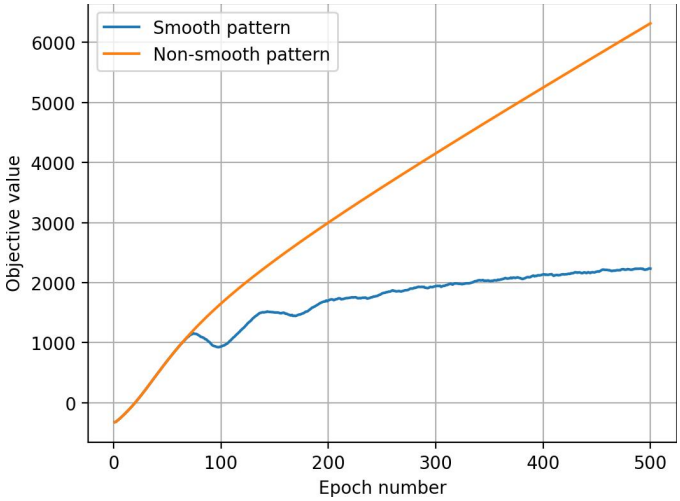


Fig. 5: The objective function when requiring smooth

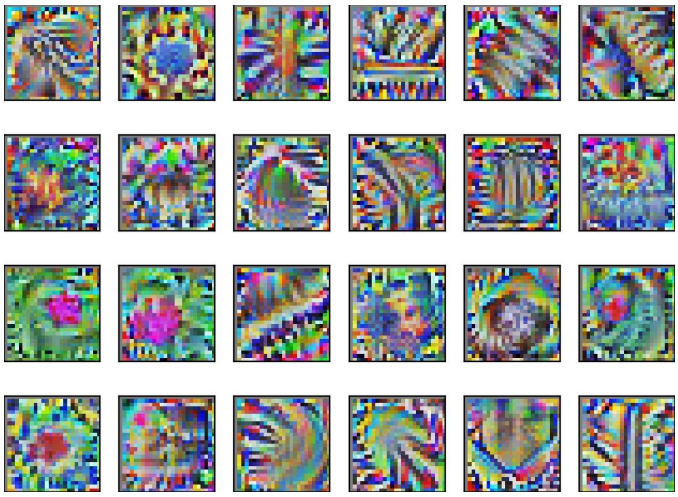


Fig. 6: Non-smooth activation patterns.

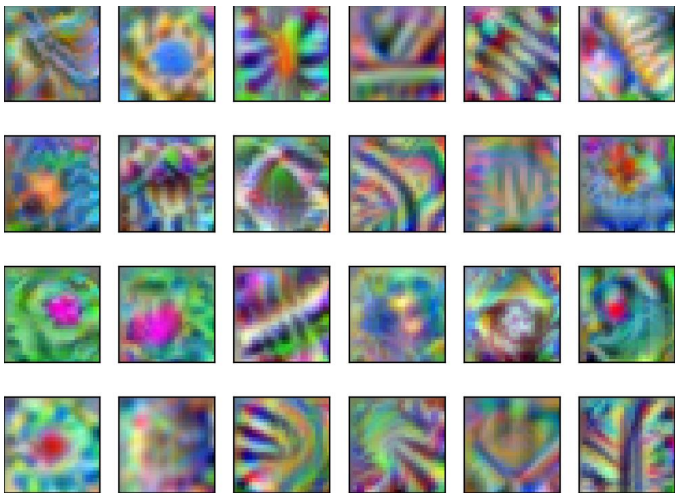


Fig. 7: Smooth activation patterns.

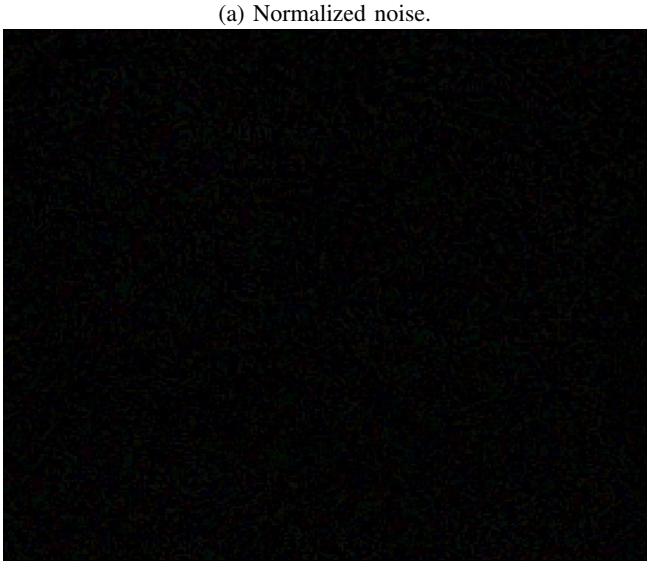
(a) Clean image

(b) The image distorted by noise.

ε	Noise mean μ	Noise standard deviation σ
0.001	(1.98e-06, -3.07e-06, -5.01e-07)	(6.45e-04, 6.74e-04, 6.08e-04)
0.01	(7.59e-06, 7.81e-06, -1.06e-05)	(2.23e-03, 2.25e-03, 2.22e-03)
0.03	(6.63e-05, -8.62e-05, 2.02e-05)	(1.74e-02, 1.76e-02, 1.73e-02)
0.1	(1.87e-04, -2.52e-05, 4.32e-05)	(3.19e-02, 3.21e-02, 3.18e-02)

TABLE II: Parameters of the noise required for adversarial attack

(a) Normalized noise.



(b) Non-normalized noise.

Fig. 9: Noise used for the adversarial attack for $\varepsilon = 0.1$.

adversarial attack was performed with several values of the hyperparameter ε denoting the radius of the l_{inf} norm ball around the clear image x_0 that the distorted image x was restricted to. When given sufficient time (amount of optimization epochs), the attack was successful for any $\varepsilon \geq 0.1$.

As the optimization algorithm runs, the classification confidence corresponding to the true class shrinks and it eventually ceases to be the most probable class. On the other hand, the true class starts moving up the order of predictions until it eventually surpasses the true class. When one probability rises, the other one drops. This process is illustrated in Fig. 10 through 13. One can immediately notice that different restrictions on noise (imposed via ε) vastly influence the true/target class probabilities/indices – with very small $\varepsilon \approx 0.001$, it may not be possible to achieve the goal. On the other hand, when the noise allowed is larger ($\varepsilon \approx 0.1$), the model converges quickly to almost 100 % certainty that it is looking at a wall clock.

The histogram 14 shows that no intermediary class was predicted when switching from the true class to the target, i.e. the transition was instant. The time of the transition may differ based on the hyperparameter – the Fig. 15 shows that when the attack has to work with a smaller ε (restrictions on the additive noise are stricter), it may take more iterations to reach a solution that confuses the model and makes it yield the target class.

Even though many experiments were performed specifically attempting to observe such phenomenon, no selection of the target class and ε lead to a case when the predicted class did not change from the true class to the target class directly

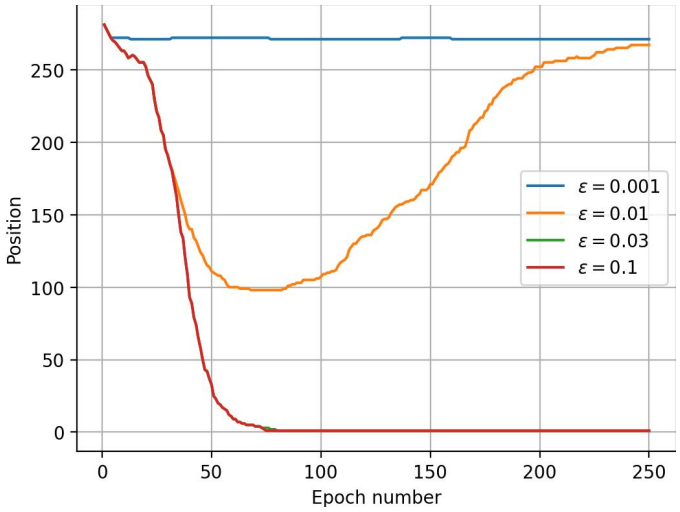


Fig. 10: Evolution of the position of the target class (wall clock) in the list of predicted classes.

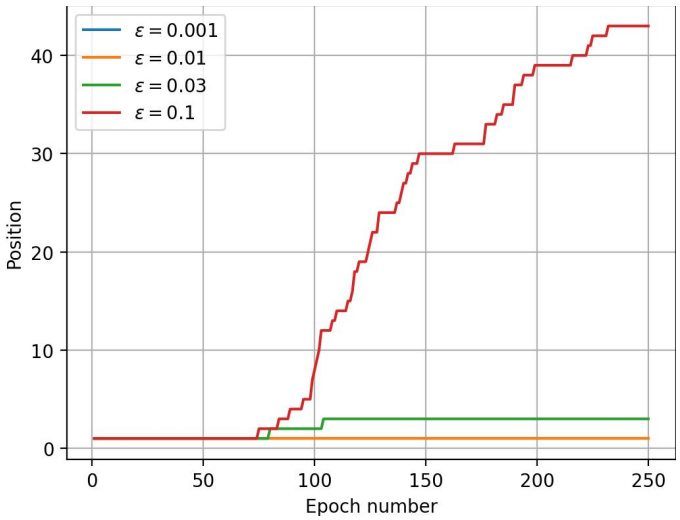


Fig. 11: Evolution of the position of the true class (Labrador retriever) in the list of predicted classes.

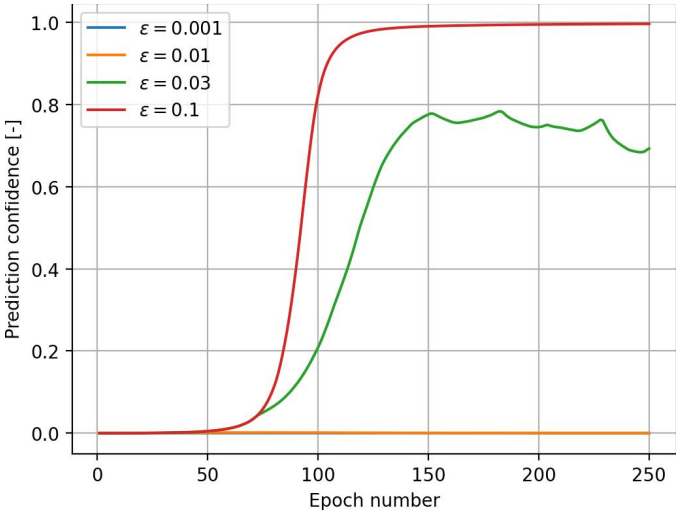


Fig. 12: Evolution of the classification confidence of the target class (wall clock).

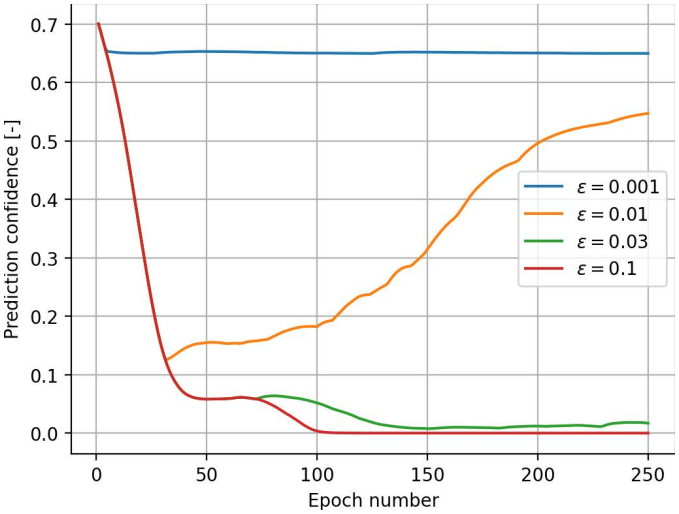


Fig. 13: Evolution of the classification confidence of the true class (Labrador retriever).

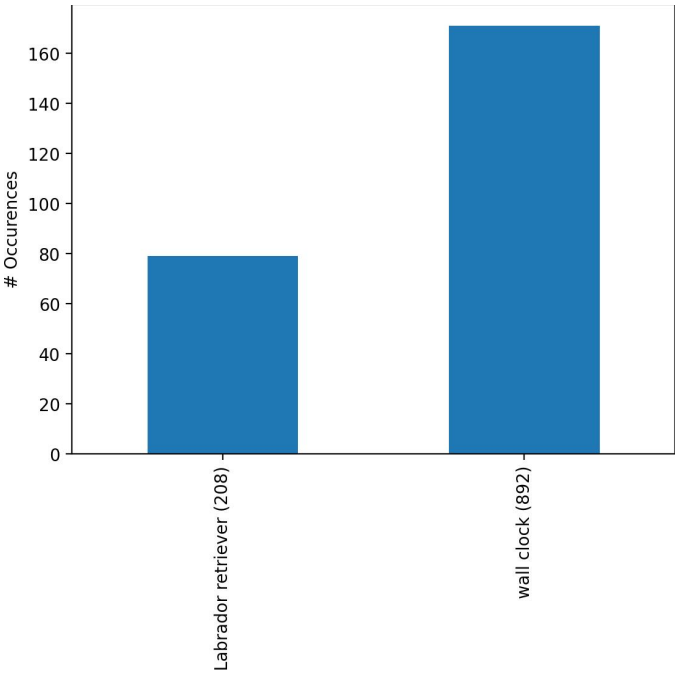


Fig. 14: Classes predicted during the attack with $\varepsilon = 0.1$.

but rather via an intermediary (transition) class. This is an indication that the iterative optimization algorithm is by no means a guess-and-check method generating random noises, but rather a systematic method trying to achieve the given goal. It should be noted that larger steps taken by the optimization with higher learning rate l_r should be more likely to trigger such event.

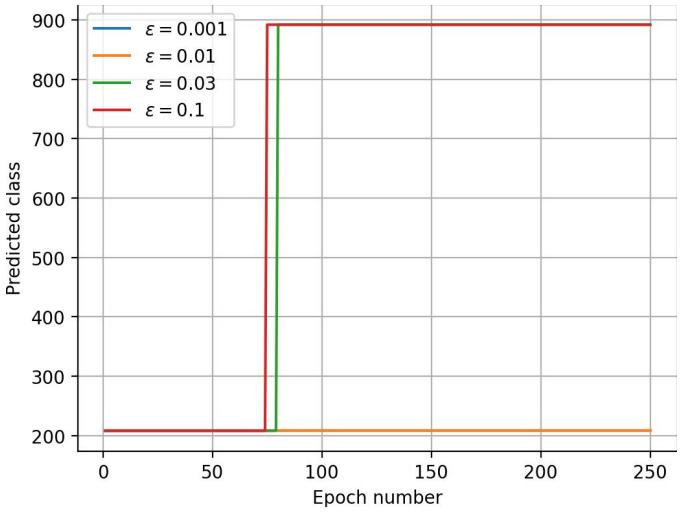


Fig. 15: Classes predicted during the adversarial attacks.