Deep Hallucination Classification

Classify images hallucinated by deep generative models

DESCRIPTION OF THE METHODS USED

A modified VGG16 convolutional network is used. The architecture used is described in **Figure 1**.

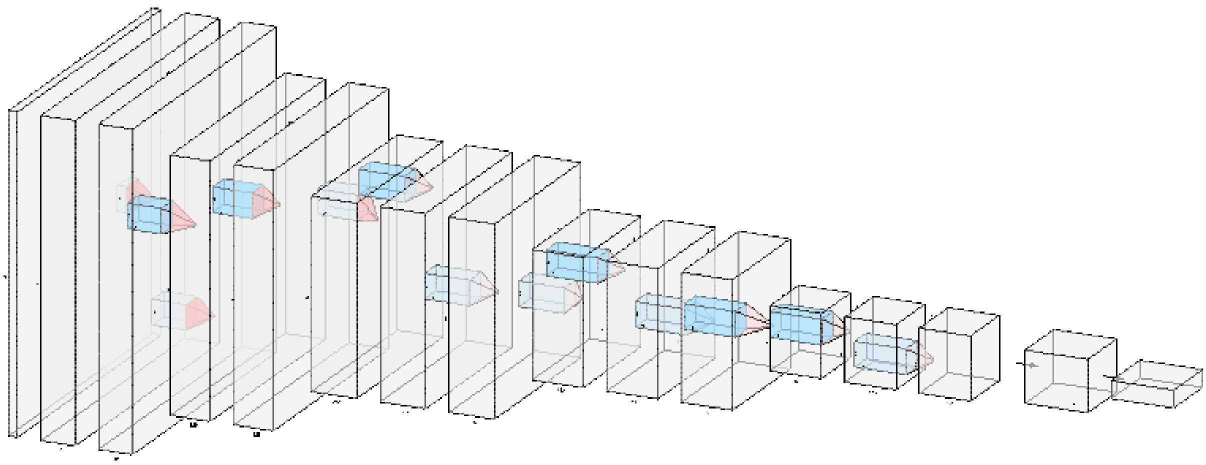

Figure 1

A detailed description of the architecture is:

a. The input data layer, which represents a 32×32 color image with 3 channels;
b. A convolutional layer with 64 size filters (3×3), padding of(1×1) and step 1 with the ReLU $(LeLU(x) = \max\{0, x\})$ activation;
c. A layer identical with b;
d. A layer of max-pooling in which the subzone in which the max-pooling is applied has the size of (2×2) with strides of 2 and dilation 1, after this step images with the size of $(16 \times 16 \times 64)$ are obtained.
e. 2 layers identical to layer b. but with 128 filters;
f. A layer of max-pooling in which the subzone in which the max-pooling is applied has the size of (2×2) with strides of 2 and dilation 1, after this step images with the size of $(8 \times 8 \times 128)$ are obtained.
g. 3 layers identical to layer b. but with 256 filters;

h. A layer of max-pooling in which the subzone in which the max-pooling is applied has the size of (2×2) with strides of 2 and dilation 1, after this step images with the size of $(4 \times 4 \times 256)$.are obtained.
i. 3 layers identical to layer b. but with 512 filters;
j. A layer of max-pooling in which the subzone in which the max-pooling is applied has the size of (2×2) with strides of 2 and dilation 1, after this step images with the size of $(2 \times 2 \times 512)$ are obtained.
k. 3 layers identical to layer b. but with 512 filters;
l. A layer of max-pooling in which the subzone in which the max-pooling is applied has the size of (2×2) with strides of 2 and dilation 1, after this step images with the size of $(1 \times 1 \times 512)$ are obtained.
m. A dense layer of 512 receivers representing the vectored $1 \times 1 \times 512$ image, with the ReLU activation function;
n. The final layer of 8 receivers to which the softmax function is added.

RESULTS

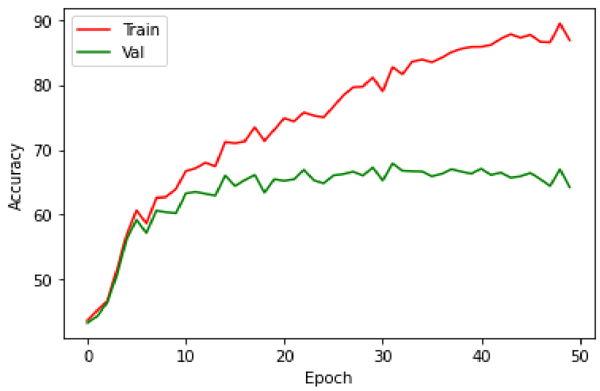

Figure 2

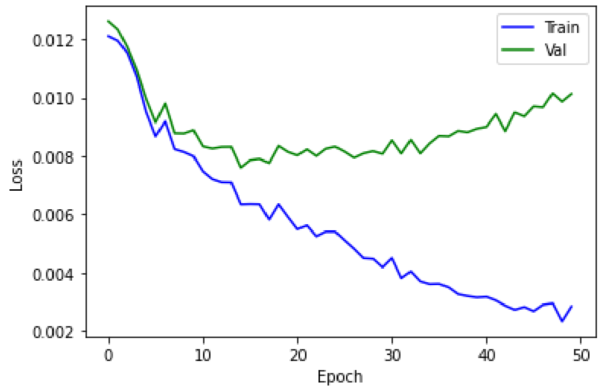

Figure 3

Figure 2 shows the evolution of the loss function, **Figure 3** shows the evolution of accuracy, and **Figure 4** the confusion matrices for the model driven by 50 epochs, with crossentropy as a loss.

$CrossEntropyLoss(x, l) = -\frac{1}{N}\sum_{i=1}^{N}[y_n log\widehat{y_n} + (1 - y_n)\log(1 - \widehat{y_n})]$ (where N the number of examples in the batch, in our case 256,$\widehat{y_n} = f(w \cdot x_n)$).

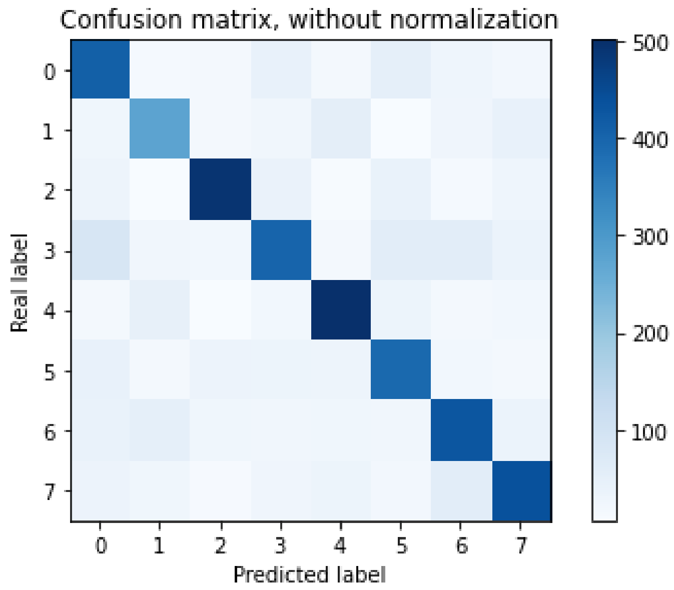

Figure 4

The accuracy on the train set is 77.76%, on validation 66.80% and on the test set (Kaggle) 67.22%. The gradient descent algorithm (SGD - stochastic gradient descent) was used with learning rate of 0.01, momentum of 0.9 and weight_decay = 0.0005.

DATA AUGMENTATION

Data augmentation was also used: for the train - RandomRotation(10), RandomCrop(32, padding=4), RandomHorizontalFlip(),ToTensor() and Normalize((0.4092, 0.4594, 0.4538), (0.2346, 0.2431, 0.2467))

And for validation and testing only ToTensor() and Normalize((0.4092, 0.4594, 0.4538), (0.2346, 0.2431, 0.2467)).

In **Figure 5,** we have a series of data after augmentation.

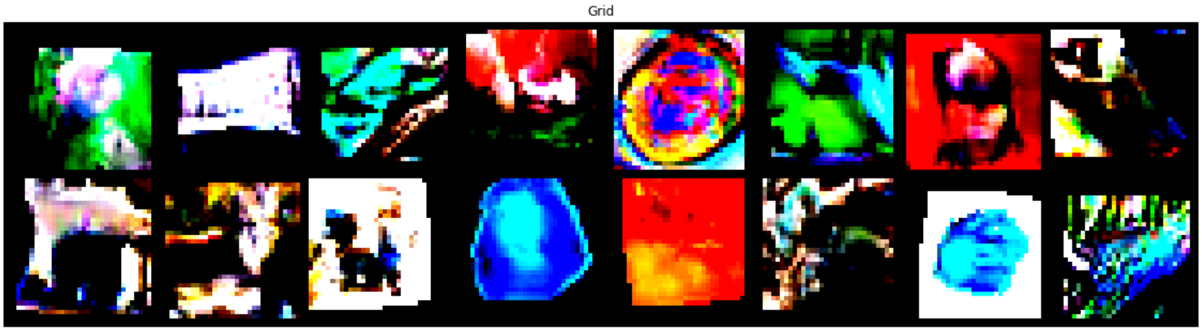

Figure 5