Assignment 4

**Task 1 - CIFAR-10 classification**

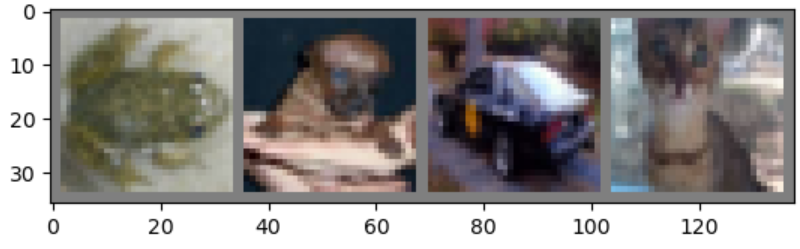
In order to learn the CIFAR-10 dataset, we followed the tutorial and built a network that consists of 2 convolutional layers, and 3 fully connected layers –



Images from the CIFAR10 are 32 x 32 x 3 images of each of the following categories –

plane, car, bird, cat, deer, dog, frog, horse, ship, truck.

Example images from the training set and their labels -



Real labels:  frog  dog   car   cat

Model training

To train this model we will use CrossEntropyLoss as our criterion and SGD as our optimizer.

[1,  2000] loss: 2.221

[1,  4000] loss: 1.826

[1,  6000] loss: 1.668

[1,  8000] loss: 1.584

[1, 10000] loss: 1.557

[1, 12000] loss: 1.488

[2,  2000] loss: 1.426

[2,  4000] loss: 1.401

[2,  6000] loss: 1.391

[2,  8000] loss: 1.358

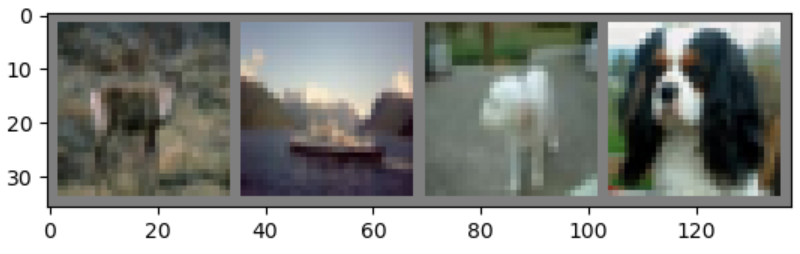
[2, 10000] loss: 1.323

[2, 12000] loss: 1.345

We can see the loss is getting lower the further into training we go.

Testing the model

Example images from the test set and their labels -



Real labels:  deer  ship  dog   dog

Predicted:  frog  ship  dog   dog

we can see that most of these the network predicted correctly and the one it failed on is somewhat hard to see even for a human without knowing it’s a deer to begin with.

Accuracy of the network on the 10000 test images: 52.8 %

Accuracy for class: plane is 46.9 %

Accuracy for class: car   is 74.1 %

Accuracy for class: bird  is 29.3 %

Accuracy for class: cat   is 21.0 %

Accuracy for class: deer  is 29.3 %

Accuracy for class: dog   is 57.9 %

Accuracy for class: frog  is 74.4 %

Accuracy for class: horse is 59.5 %

Accuracy for class: ship  is 82.8 %

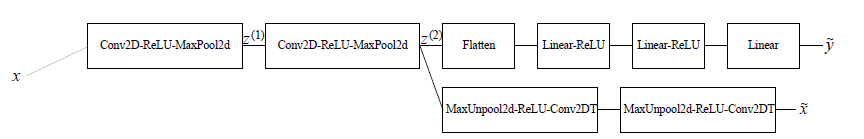
Accuracy for class: truck is 52.8 %

The model achieved 52.8% overall accuracy. Compared to a random guess between the 10 classes (which would have given us 10%) we can say that the model has learnt to classify the dataset.

We can also see that on some classes like ship or frog the model did very well while on classes like cat or deer it did worse.

**Task 2 - Deconvolutional Model**

In order to reconstruct images after our convolutional layers, we adapted our previous network to include 2 new deconvolutional layers and a new output of the reconstructed image.



The new loss for this network will be a combination of the old CrossEntropy and mean squared error (MSE) of the original input and its reconstruction.



We will also have a new hyperparameter for the network that will decide which part of the loss has more weight. The higher the value of is, the better the reconstruction should look but the prediction might not be as good. The same goes for the other way around.

Hyper parameter testing

To decide on the right value of we tested several values to see which reconstructed images look the best while making sure the accuracy doesn’t drop too much.

We decided on = 2 since its images had most of the details while also not hurting the accuracy.

Images of the tests with different values can be found at the end of the report under “Task 2 - hyper parameter testing”

Label prediction

After retraining the new network with the new deconvolutional layers we get the following accuracies -

Accuracy of the network on the 10000 test images: 53.8 %

Accuracy for class: plane is 61.3 %

Accuracy for class: car   is 78.6 %

Accuracy for class: bird  is 38.0 %

Accuracy for class: cat   is 22.4 %

Accuracy for class: deer  is 44.6 %

Accuracy for class: dog   is 26.2 %

Accuracy for class: frog  is 64.1 %

Accuracy for class: horse is 77.9 %

Accuracy for class: ship  is 65.8 %

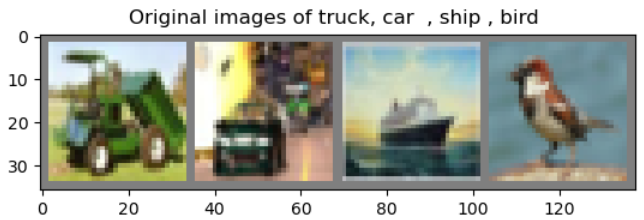
Accuracy for class: truck is 59.1 %

We can see that the accuracies have not changed much compared to training it without the deconvolution layers, showing that we didn’t hurt the ability of the network to classify images.

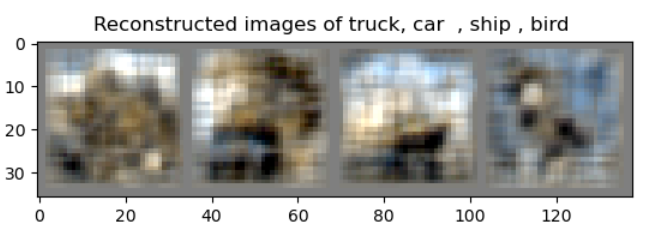
Image reconstruction

Here are some examples and their reconstruction.

Original images from the test dataset and their labels -



Same images reconstructed after two convolutional layers and two deconvolutional layers -



Comparing each image to its reconstructed counterpart, we can see that most of the image’s features are still in place: the outlines of the main thing in the image, the colors, the scale and the orientation.

While the reconstructed images are still blurry (even more than before), you can see the resemblance between them and the originals, and even guess what the label of the some of the images is.

**Task 3 - Latent Representations Analysis**

In order to see how the network learnt to classify images we will visualize the Latent Representations the network has learnt.

We took two images, one from the test dataset and one from the train dataset, fed them through the network and reconstructed them each time with only one of their channels (the rest were zeroed).

The following table will show the different images that were reconstructed along with our guesses about the patterns arising from these images.

|  |  |  |
| --- | --- | --- |
|  | Train Image | Test Image |
| label | plane | horse |
| Original Image |  |  |
| Conv1 – Channel 0 |  |  |
| Conv1 – Channel 1  Highlight dark colors in the image |  |  |
| Conv1 – Channel 2  Areas with bright/ white colors in the images |  |  |
| Conv1 – Channel 3  Edges or specifically the color blue (to see where there is sky or water) |  |  |
| Conv1 – Channel 4  Similar to channel 3, maybe marking what is background and what isn’t. |  |  |
| Conv1 – Channel 5  Points of interest, the area where the main thing in the image is and its general shape |  |  |
| Conv2 – Channel 6  The general shape of the plane can be seen but with what seems to us as noise |  |  |
| Conv2 – Channel 12 |  |  |
| Conv2 – Channel 15 |  |  |
| Fully reconstructed |  |  |

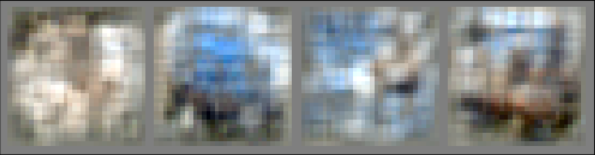
In general, we can see that each channel learns some specific information about the image, even if its hard for us to see and understand exactly what it is. Maybe with more images we could tell what patterns are visible in each channel better.

On top of that, we can also see that the channels in the first layer are very simple with little details, which made it easier to point out patterns in them, while the channels from the second layer have much more details and differing colors in them. This conveys the idea that each layer of the network transforms the input data into a more abstract and higher-level representation. These representations might not be directly observable or interpretable by humans but are usefull for the to the networks ability to classify images.

Task 2 - hyper parameter testing



= 0.5 

= 1 

= 1.5 

= 2 

= 3 

= 10 