**Convolutional Neural Networks for Multi-class**

**Histopathology Image Classification**

**Students: Noy Maman, Asaf Darmon, Karin Azulay, Liran Hersh.**

**The process of the project study**

In this project, our task was to diagnose diseases via histopathological images using ResNet.

At the beginning, we searched about neural networks to learn the background of the subject we are dealing with. We started to understand how neural networks and CNN algorithms work and then moved to ResNet and ResNet50.

CNN contains layers which includes two basic layers named convolution and pooling that helps analyze images input.

During our learning process we were exposed to a lot of information from the article and across the internet. Due to our learning process, we discovered the whole architecture of ResNet.

During the work process we took a data set of images and an excel document which helped us to train our network.

At the beginning, we used a smaller image data set and saw that as we increase the amount of images we get greater accuracy and a suitable result, therefore we used 10,020 tissue images in our project.

Finally, we were able to train our network and to diagnose cancer disease via histopathological images using ResNet50.

**project flow**

* Data Base – took a Data Base which contains 10,020 tissue images and excel document with all the images and labels (0 and 1) which indicates whether the image is cancer or not.

At the beginning we loaded 6000 tissue images and saw that the accuracy we were able to achieve is 74%, therefore we decided to increase the number of images to 10,020 so that will assure us a higher accuracy of 80%.

The training time was much longer but, in that way, we got the desired result of higher accuracy. In our Data Base we also have an excel document (train\_labels.csv) which contains all the images names next to respective label that indicates whether it is cancer or not.

* "Preparations" - Load and display some positive and negative images, Split the train data into train set and validation set and create a dataframe which contains every training path, id and label.

Then we choose 4 random positive and negative images, find their respective path then and display them in a subplot 2x4.

* Validation – split 20% of the training set into a validation set.

On this way the network can test itself if the results are reliable.

* Loading the network –

conv\_base = ResNet50(weights = 'imagenet', include\_top = False, input\_shape = (IMG\_SIZE,IMG\_SIZE,3))

At this line conv\_base contain a call for ResNet50 function that includes all the layers and pooling with input images of size 196x196x3, where 3 is the number of color channels (r,g,b).

* Training –

1. Change ResNet50 standard training on ImageNet to our specific training set. Because of that, we will need to train the last few layers instead of the just the last one.
2. Choosing optimizer - An optimizer is one of the two arguments required for compiling a Keras model. optimizers.Adam(0.001).
3. Select the number of images to train, an early-stopper and a reduce which are two options to choose from if learning rate stops improving, usually in the last epochs.
4. Start network training and validation with 10 epoch and train\_step\_size (which is the number of steps per epoch) as we calculated before.

(train\_step\_size=train\_generator.n //train\_generator.batch\_size).

Because the data set is big, training is slow.

* Graphs – set up and design the result graphs.

Now that our model has been trained, it is time to plot some training graphs to see how our accuracies and losses varied over epochs.

**The obtained result**

The ResNet50 accuracy we got is 80%.

**Conclusions**

It should be noted that all the tissue images in this database are very similar images so the accuracy we got is relatively good and we can say that our network was diagnosed cancer disease successfully.

Because we used our home PCs, we used a very small sets of images, (10,020 instead of 220,000 ) and got 80% accuracy.

If we were use the whole images, we could get much bigger accuracy.