PhD Progress: Development and application of deep learning methods for medical image analysis
Ανάπτυξη και υλοποίηση μεθοδολογιών βαθιάς μάθησης στην ανάλυση ιατρικής εικόνας

George Xenogiannopoulos

Biomedical Engineer

M.Sc. in Advanced Electronics Engineering

February 2022

1 Deep Learning Algorithms

The second year of this PhD study aims to integrate deep learning tools in Medisp Ml platform. Since the platform was built in Python we used the TensorFlow[1] library for deep learning operations and methods. Trying to solve classification problems on medical images, we build a pipeline that trains a supervised learning model to be able to classify images of two or more classes. So far we have developed a basic Convolutional Neural Network (CNN) architecture, based on Keras[2] as well as we have integrated half a dozen pre-trained models available through TensorFlow Hub[3]. Following the Neural Network architectures we have integrated so far in Medisp Ml platform:

1. Convolutional Neural Network (CNN)
2. Visual Geometry Group 16 (VGG16)
3. MobileNet Version 2 (MNV2)
4. Inception (INC)
5. ResNet50 (RES)
6. EfficientNet (EFI)

Of course the above list can very easily be extended with any available network available in TensorFlow Hub[3]

2 Materials and Methods

2.1 Importing data to Medisp ML platform

We chose a dataset that was published by Warwick University for GlaS@MICCAI'2015: Gland Segmentation Challenge Contest[4]. The first publication of this dataset was from Sirinukunwattana and his team where they developed a segmentation algorithm named Random Polygons Model (RPM) for extracting glandular structures from histology images of colon tissues [1]. However, we will use this dataset to try to solve classification problems rather than segmentation problems. The dataset consists of two categories and 165 images in total. All the images are RGB colored and vary in height from 422 to 522 pixels and in width from 574 to 775 pixels. From the whole dataset, 74 of the images belong to the benign category while the rest 91 belong to the malignant category. For every one of the dataset images there is a mask that annotates the area of the gland. A sample from each class can be seen below in Figure 1 along with their masks in Figure 2.

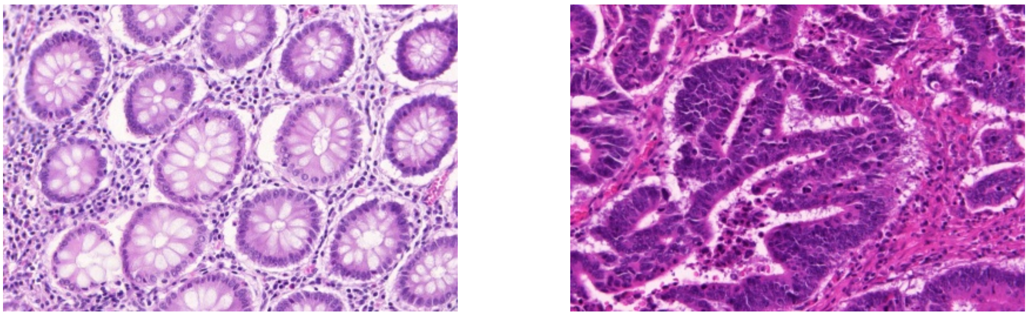
[1] `https://www.tensorflow.org`

[2] `https://keras.io/`

[3] `https://tfhub.dev/`

[4] `https://warwick.ac.uk/fac/cross_fac/tia/data/glascontest/`

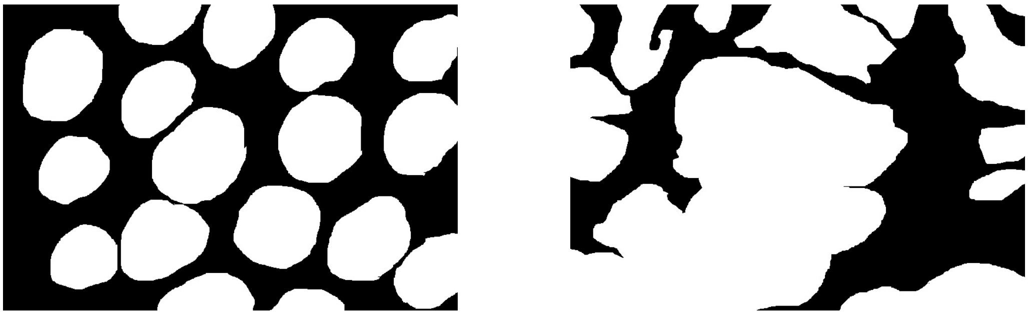
Figure 1: Histologic Samples



(a) Benign

(b) Malignant

Figure 2: Mask Samples



(a) Benign

(b) Malignant

Only labeled datasets can be imported to Medisp ML platform, so far, with the following folder structure. The root folder should contain one folder for each category, named after the category's name. The category folder should contain all the category's images. In case mask annotations are given, they should also be placed inside the category's folder. The annotation files should have exactly the same name with the corresponding image. All the annotation files as well as a suffix or prefix that can be used as mask indicator when iterating

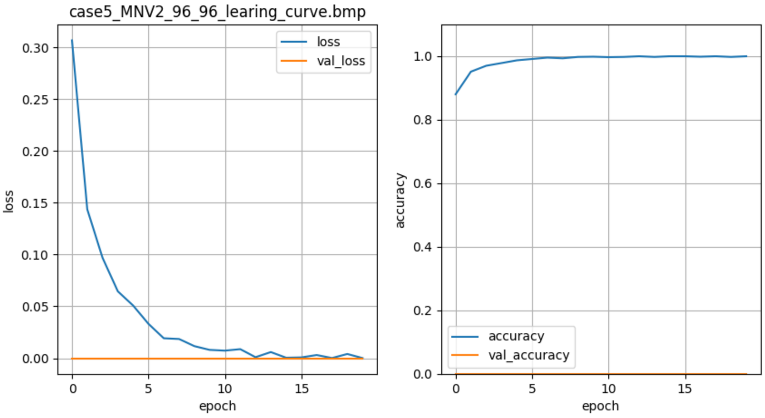
2.2 Training a Deep Learning model

When not many labeled samples are available, as in our case, transfer learning techniques provide better results than training a network from the scratch [2]. We tried training a model from scratch and also pre-trained models. Taking into account the small size of our data set, we tried three different approaches, and we compared the results. All the pipelines start by splitting the whole set of the images into two sets. The first set is going to be used in the training process, and we call it the training set. The second one will not participate at all in the training process and will be used later for validating the model's performance, and we call it the external set. In particular we will employ the k-fold technique, where the dataset is split into K-folds (e.g., 3 folds), 2 folds are used to train the network and the one fold is held out to be used as the external set of images for classification. Next, the second fold is held out and the remaining 2 folds are used for training the network. By repeating this cycle 3 times, one for each fold, every image is classified by the network that has been trained by different images to the ones used for classification. By the end of the process we calculate the overall score of the model by counting the correctly classified images and this we call prediction. In addition, we record the minimum loss and maximum accuracy of the model during the training phase.

Train using the whole image

Our first attempt was the simplest one, to train the network using the whole images. The pipeline starts by splitting the dataset into training (2-folds of images) and external sets (1-fold of images). The models adjust the weights and label the images, producing in this way the training metrics loss and accuracy. From those epoch evaluations we generate the learning curve of the model which can be seen in Figure 3. After the training phase finishes, we use the trained model to predict the external set and this is how we evaluate the total performance of the model.

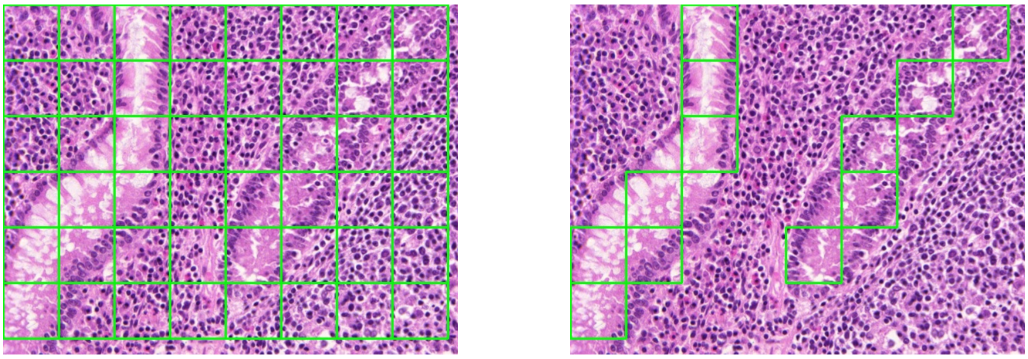
Figure 3: Learning Curve



Train on image patches

Deep learning networks need to be trained on a large number of samples in order to ”gain knowledge”. In order to increase the number of samples and study the model’s performance, we cropped the images into patches. Each patch inherits the labels of its parent image and this time we train the model on patches rather than the whole image. The patch size can be customized by the user, however we did use the maximum of the minimum image shape that the pre-trained networks that we use can accept as input. The pipeline starts by splitting the whole dataset into training and external sets. All images, from both sets, are then cropped to patches of the given input size. We choose (96, 96) input patch size, for most of the networks, which is the minimum input size of MNV2-network. Training on EFI-network was made with input sizes (224, 224). Figure 4a visualizes the result. After the training phase completes, we use the trained model to predict the external patches. The external images are then predicted to the class in which the majority of its patches are classified.

Figure 4: Patch Samples



(a) Patches without mask

(b) Patches with mask

Train on image patches extracted from mask area

To further improve the second approach, we discarded the patches located outside the gland using the mask annotations, see Figure 2. The pipeline begins as usual, the first step is to split training and external sets. In this approach we skipped training on EFI since its minimum input size is (224, 224) which is much larger than the glands areas inside the images. Then all images, from both sets, are cropped to patches with the additional step that patches with less than 70% mask coverage are discarded. Figure 4b visualizes the result. The training and prediction phases are similar to the previous approaches, after training on patches, we predict external patches and classify the image according to majority. By discarding patches outside the region of interest, on one hand reduces the number of samples but on the other hand we train the network only on the areas of differentiation. We expected this approach to improve the overall performance of the models. The filtering of patches outside the gland annotation, acts like removing the noise from the training phase. At this point a big discussion can be made on whether this technique helps the network to generalize or leads to overfitting [3].

3 Results

Train using the whole image

The test run with the following parameters:
image height = 224, image width = 224, epochs = 20, folds = 3

Network	Loss	Accuracy	Prediction (%)
CNN	0.548	0.758	55.15
VGG16	0.005	1.0	94.55
MNV2	0.001	1.0	97.58
RES	0.0	1.0	81.21
INC	0.0	1.0	93.33
EFI	0.0	1.0	94.55

Train on image patches

The test run with the following parameters:
patch height = 96, patch width = 96, epochs = 20, folds = 3

Network	Loss	Accuracy	Prediction (%)
CNN	0.32	0.842	79.39
VGG16	0.109	0.953	96.36
MNV2	0.012	0.995	96.97
RES	0.015	0.992	93.94
INC	0.021	0.996	94.55
EFI	0.011	0.995	94.55

Train on image patches extracted from mask area

The test run with the following parameters:
patch height = 96, patch width = 96, epochs = 20, folds = 3

Network	Loss	Accuracy	Prediction (%)
CNN	0.326	0.839	81.82
VGG16	0.035	0.989	97.58
MNV2	0.0	1.0	97.58
RES	0.001	0.999	96.97
INC	0.004	1.0	97.58
EFI	-	-	-

Short discussion on the results

According to the results the following statements can be made:

- We verified that pre-trained networks perform better than models trained from scratch, especially on small datasets.
- Predictions when using patches score higher than the model's external evaluation due to the majority vote on patches classification.
- Results verified that our attempts to increase overall accuracy were in the right direction. Using patches performs better than not using and discarding patches outside the region of interest performs better than including all the image patches.
- The main drawback is that mask annotations are not always available. A segmentation model that can quickly extract the glandular structure, even if it is not very accurate, can help our technique to perform on new data very accurately.

References

[1] Korsuk Sirinukunwattana, David R. J. Snead, and Nasir M. Rajpoot. A stochastic polygons model for glandular structures in colon histology images. *IEEE Transactions on Medical Imaging*, 34(11):2366–2378, 2015.

[2] V. García-Díaz. *Algorithms in Decision Support Systems*. MDPI AG, 2021.

[3] R.S. Mangrulkar, A. Michalas, N. Shekokar, M. Narvekar, and P.V. Chavan. *Design of Intelligent Applications using Machine Learning and Deep Learning Techniques*. CRC Press, 2021.