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 2024

1 Introduction

Having reached the third year of this study, we begin with a short review of the previous years and the development of the system and the experiments so far. During the first year, we built the infrastructure and the core code of the web-based application called "Medisp ML" that we are using to do all of our experiments for this PhD research. To perform discriminant analysis, we build a pipeline that combines different classifiers, different feature reduction methods, and various methods for evaluating results. We used scikit-learn and TensorFlow python packages for this purpose. We chose image analysis features extracted from CRC (Colorectal Cancer) histologic images as train and test datasets. Finally, we tested the machine learning pipelines and we reported the accuracy scores for different classifiers on our datasets. For more details refers to the 1st progress report by clicking the link. The second year's main focus was to build a second classification pipeline based on deep learning methods in order to compare results with the discriminant analysis pipeline. We used TensorFlow and Keras ML libraries to build our deep learning pipelines. We experimented with different approaches to train and evaluate various neural network architectures on the same dataset of CRC images that we also used in the first year. One expected outcome of the second's year study was that pre-trained networks performed much better than models trained from scratch, especially on small datasets [1]. The other interesting finding was that training the models on small parts of an image, called tiles or patches, rather than the whole image, improved the overall accuracy of the models. To explain further, the image's classification was based on the most common prediction among all the tiles. We also observed that the model's accuracy improved when we cropped the tiles only from the regions of interest, which in this case were the glandular areas. For more details refers to the 2nd progress report by clicking the link.

2 Deep Learning fine tuning

The third year of this PhD study aims to explore on ways to increase efficiency of the deep learning models, that have already been utilized in effectively classifying the colorectal histological images data set that was published by Warwick University for GlaS@MICCAI'2015: Gland Segmentation Challenge Contest[1]. Starting by studying the literature for common practises to improve classification results in small training data, the first attempt to our goal was to apply data augmentation in training phase. Data augmentation is a widely used technique to improve overall performance of classification models by generating variety of diverse images during the training and testing process. This technique is very useful especially when the number of training data is relatively small [2]. Using data augmentation we managed to overcome the problem of our small data set which makes the deep learning models difficult to train. Furthermore we integrated the ensemble classification technique in order to increase accuracy and prediction reliability. We achieved that by training multiple different models and using their predictions in voting mode [3]. Later we split the dataset into smaller

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

subsets, and we combined those subsets in different ways to create datasets, of two and three classes, meaningful for the physicians. While studying the literature for similar works on topics of deep learning in colorectal image classification, we came across the article of Devvi Sarwinda [4] were they actually published their work using the same dataset of the GlaS@MICCAI'2015 contest as we do. In their work they are applying an image processing method for histogram equalization, called CLAHE, in pre-training phase, in order to increase image contrast and classification results. Eventually we integrated their method to optionally into our pipeline to examine the effect. Final attempt was to build a hybrid classification model which combines a deep pre-trained and a shallow neural network to make a prediction. The input of the deep network was an image and the input of the shallow was a feature vector extracted from that image. After integrating all the techniques described above to out pipeline, we conducted fine-tuning for each model to investigate how adjustment of the parameters affect the model's adaption to each dataset.

3 System development and Methods

3.1 Split dataset

The whole data set is labeled in five categories which can be seen bellow in Table 1. In order to test the algorithms and prove effectiveness, we tried an easy problem and that was to merge the classes "Healthy" and "Adenomatous" into one named "Benign" and the rest three classes "Moderately differentiated", "Moderately to poorly differentiated" and "Poorly Differentiated" were merged into one class named "Malignant". The results of that two class problem "Benign" VS "Malignant" can be found in our previous report in this link. Distinguishing an image between benign and malignant is not a challenging problem for a physician, however distinction between different grades among the malignant ones or between resemblance categories could be more intriguing. We tried different ways to create datasets of two and three classes. We did this by merging

Label No	Label	Category	Number of samples
1	Healthy	Healthy	42
2	Adenomatous	Benign	32
3	Moderately differentiated	Grade I	47
4	Moderately to poorly differentiated	Grade II	20
5	Poorly differentiated	Grade III	24
			Total: 165

Table 1: Detailed description of the GlaS@MICCAI'2015 contest dataset regarding the labels, the grading and the number of images per class.

the labels in a way that is meaningful for the physicians or a case study. Having Grade II and Grade III with fewer labeled data than Grade I, and actually almost the double, it would be reasonable to merge Grade II + Grade III into one class and compare against Grade I and this is how Set number 1 was created, a two class problem with balanced samples. Set number 2 was created for for symmetry reasons which is also a two class set where this time Grade I + Grade II labeled items were merged into one class against Grade III labeled items. Even thought we do not anticipate favorable outcome because of the imbalanced number of samples between the classes we decided to endeavour the task. Sets number 1 and 2 are also described in Table 2. Having examined only binary problems so far, we extended the code to also support multi-class classification problems and tried with a couple of three class sets. The first one, set number 3, can be considered an easy one because the classes healthy, benign and malignant naturally diverse. Malignant class consists of all Grade I, Grade II and Grade III labeled samples. Finally we wanted to examine whether the three malignant classes Grade I, Grade II and Grade III can be distinct by out system and this how set number 4 was created. Sets number 3 and 4 are also described in Table 3. All the subsets details described above, can be seen below in the following tables together with the number of samples per class.

Set No	**Class 1**	**Class 2**	**Samples**
1	Grade I	Grade II + Grade III	47 vs 44
2	Grade I + Grade II	Grade III	67 vs 24

Table 2: Two class sets of combined categories.

Set No	Class 1	Class 2	Class 3	Samples
3	Healthy	Benign	Malignant*	42 vs 32 vs 91
4	Grade I	Grade II	Grade III	47 vs 20 vs 24

*Malignant class is a merge of Grade I, Grade II and Grade III categories.

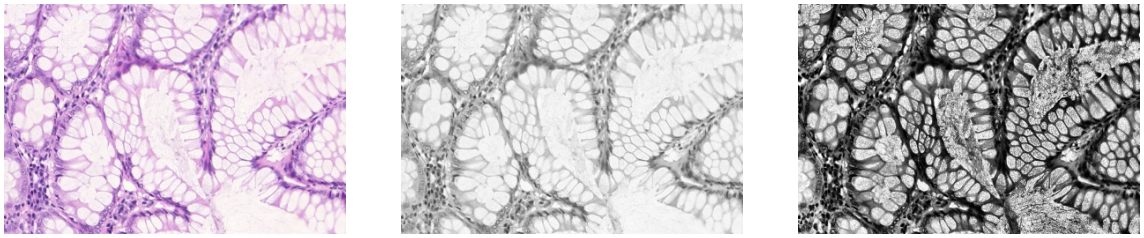
Table 3: Three class sets of combined categories.

3.2 Ensemble classification

Ensemble classification is technique widely used in machine learning which aims to increase overall results by taking into account the outcome of more than one classifier before classifying a sample [3]. The way we implemented this method was to train more than one of the available models with the same data in each fold iteration. Then, in each fold, all the models predict the probability of a sample to belong to a class, then the probabilities are summed per class and the sample is assigned to the class with the highest probability sum. We decided to predict the probability of a sample using the sparse categorical cross-entropy output of the loss function and not to predict the label directly for two reasons. The first reason is because each sample may belong to exactly one class, this can be also called that the classes are mutually exclusive. The second reason is more practical and is that when predicting class of a sample from an even number of classifiers or for more than two classes, counting the majority vote can be impossible.

3.3 CLAHE histogram equalization

CLAHE is acronym of Contrast Limited Adaptive Histogram Equalization. The method of Sarwinda [4] applies the filter in pre-processing phase in order to increase image contrast and possibly the classification results. It was debatable whether to apply the CLAHE filter on the whole image and then extract patches or to apply the filter autonomous on each patch. We finally decided to apply the filter on the patches because we were interested in enhancing contrast locally depending on the patch content. However the CLAHE method was designed to operate in small tiles of the image and this is how it manages to adapt to different local histogram imbalances [5]. The final factual statement indicates that applying CLAHE on patches achieves maximum contrast increment.



(a) Color-scale RBG (b) Grey-scale conversion (c) CLAHE equalization

Figure 1: Steps of image transformation when applying CLAHE.

3.4 Hybrid model

The hybrid model is a concatenation of a machine learning and a deep learning models. A block diagram of it's architecture can be seen below in Figure 2 The idea is to create an image classification model that accepts as input the features extracted from the output of a pre-trained model combined with the image analysis textural features extracted from the image. The hybrid model accepts two inputs, a three-channel image and a feature vector. The features have been extracted for each image, or image patch, prior to model training phase. The feature list can be found in the Appendix A. The first layer of the hybrid model consists of a Keras pre-trained model and a simple three-layer neural network. The Keras pre-trained models can be any of the list in the Appendix B The image input is ingested into the pre-trained model of the first hybrid layer. Having replaced the classification layer with the Global average pooling operation for spatial data layer, the pre-trained model returns a feature vector. The feature vector generated from the deep learning network is concatenated with the features extracted from the image analysis algorithms. The result of the feature concatenation is what we call hybrid feature vector. The hybrid vector is the input of the second hybrid layer which is a shallow network with only one internal layer that does the final classification.

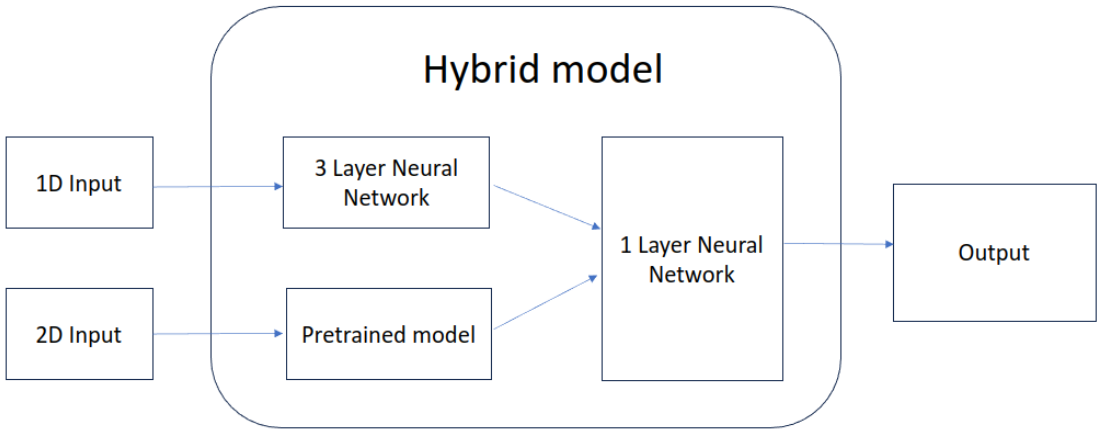


Figure 2: Block diagram representation of the Hybrid model.

3.5 Data augmentation

Deep learning models are hard to train not only due to their increased demand in computational resources, but also because they require a big amount of training data [6]. The challenge we are facing with this data-set is the training samples scarcity. Even though we are working with the patches which are considerably increased in number comparing to the images number, the actual info they are containing in number of pixels is even smaller because we only extract patches from the region of the gland and not the whole image. Furthermore, augmentation techniques are still important to be applied during the training phase because they increase the diversity of training data, help preventing over-fitting and also help balancing classes. We chose to apply data augmentation transformation that don't affect textural information on the image which is essential for medical image classification. For the reasons mentioned above we have applied the following transformations which are already integrated in Tensorflow and Keras software:

- **Flipping**: Applies horizontal of vertical flipping of the image randomly. An example can be seen in Figure 3 in the first row.
- **Random rotation**: Rotates the image by $90°$ or $-90°$ randomly. An example can be seen in Figure 3 in the second row.

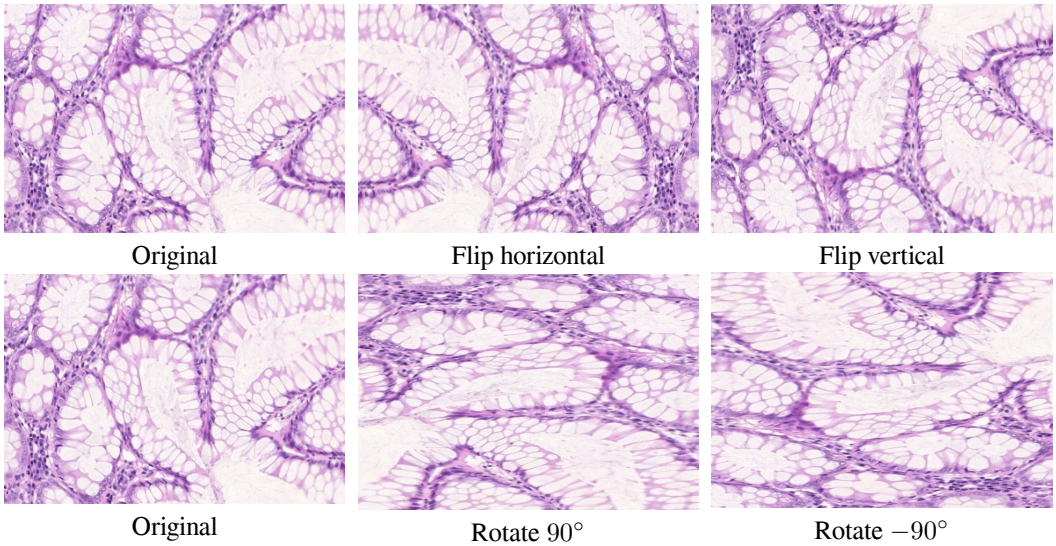


Figure 3: Representation of data augmentation transformations of an image. Horizontal or vertical flips and clockwise or counter-clockwise rotations.

4 Discussion and Results

4.1 Fine-tuning

Having integrated the methods mentioned in the development section, we run some tests to evaluate the impact of each method in model's performance, if there was any. To serve this purpose, the datasets were created with different levels of difficulties in order to challenge the deep learning networks and also our training pipelines. However, those test aim to measure whether there is any improvement in the training score and not the training score itself. The first run was to set the ground truth on how the pre-trained models perform on the new datasets by running our simple training pipeline without any enhancement methods. All the runs were executed with the following training parameters fixed.

Epochs	Patch size	Train size	Test size	K folds	Batch size
20	[78,78]	0.8	0.2	3	32

Table 4: Static training parameters for all runs.

4.2 Ensemble classification

We choose to involve in the ensemble classification only the pre-trained models with highest score in the simple pipeline between all classes. These models are from higher to lower accuracy, RES50, MNV2, EFIB0 and in the following Table 5 we present the training results with single classifier and ensemble. In all the cases the ensemble score is lower than the highest single network but higher than the average score of the individuals. This paragraph was introduced first in order to be able to include the ENSBL classifier in the rest of the results.

Set No	INCV3 (%)	MNV2 (%)	RES50 (%)	Average (%)	ENSBL (%)
1	70.33	76.92	79.12	73.99	74.73
2	81.32	78.02	84.62	79.12	81.32
3	89.7	90.91	93.94	90.91	92.73
4	62.64	65.93	72.89	66.42	67.03

Table 5: Classification accuracy for each classifier on every set with static parameters including Ensemble classification schema.

4.3 CLAHE histogram equalization

With a quick look into the results §4.7, Applying CLAHE on patches does not appear to improve the models' performance or understanding of the data. One key difference with Devvi Sarwinda's work [4] is that he applied histogram equalization on the whole image and not in patches. It looks like the significant information that helps a model to classify an image increases with an increase of contrast in the whole image and not an increase of contrast in texture details. Another reasoning that could explain the poor improvement is that CLAHE filter is only applied on grey-scale images while the pre-trained networks were trained and accept as input, color images. Providing the same gray-scale input for all three RGB channels of the network may downgrade the model's performance. It is sure that increasing contrast on patches does affect the models performance but is not always in a positive way. It is not clear how and in which case's applying CLAHE filter could be useful and would require further investigation. Some ideas for future work could be to apply CLAHE on bigger patches, perform colorful CLAHE or just training for more epochs.

4.4 Hybrid model

Summarising the results §4.8, the hybrid model seems to improve accuracy score in most of the cases. Actually in 18 out of the 28 cases the mean increase in score is 16.43% while in 7 out of the 28 cases the score was decreased with mean decrease -2.7% showing that the amount of improvement is much higher that the amount of degradation. A closer examination of the results reveals an interesting finding that the ensemble classification in combination with the hybrid model achieves the highest classification score in all the datasets and also the highest classification result overall the runs.

4.5 Data augmentation

Examining the augmentation results in §4.9, the widening of train data variety and diversity through data augmentation seems to be powerful in the models that have the lowest score on the datasets and these are CNN and VGG16. Especially in CNN, which is untrained model, augmentation techniques seem to help the model learn more effectively on our relatively small dataset, even on the difficult tasks. On the pre-trained models, and in most of the cases, we noticed a slight decrease in score and the reason for this might be the introduction of new training examples that facilitate generalization of the model. The effect of generalization would be beneficial in score if the training epochs were more and this is something that is planned to be examined in future work. Another interesting finding is that the applying data augmentation together with ensemble classification is improving accuracy on all datasets. Even if on each individual model the data augmentation causes accuracy decreases, the overall performance of the ensemble model is positively affected and actually, it achieves the highest score on every dataset.

4.6 Future work

The development of the algorithms and has reached a satisfactory stage and no additional features are planned to be added. Only needs for code optimization or minor fixes might cause additional code development. Utilizing the methodologies presented in this progress review, training processes will continue to search for them optimum parameters. Results that will be generated will be used to compose two articles for publication and submission, and one conference poster. After these tasks are completed, the remaining steps will be the preparation and defense of my PhD thesis to conclude my research.

4.7 Results CLAHE

Pre-trained model	Set No 1 Prediction (%)	CLAHE Prediction (%)	Difference (%)
CNN	46.15	51.65	11.92
VGG16	65.93	54.95	-16.65
EFIB0	65.93	64.84	-1.66
INCV3	70.33	73.63	4.69
MNV2	76.92	76.92	0.0
RES50	76.92	65.93	-14.29
ENSBL	74.74	70.33	-5.9

Table 6: Classification accuracy for each classifier on Set No 1 with static parameters and CLAHE pre-processing.

Pre-trained model	Set No 2 Prediction (%)	CLAHE Prediction (%)	Difference (%)
CNN	73.63	73.63	0.0
VGG16	61.54	67.03	8.92
EFIB0	74.73	73.63	-1.47
INCV3	81.32	80.22	-1.35
MNV2	78.02	81.32	4.23
RES50	84.63	84.63	0.0
ENSBL	89.01	81.32	-8.6

Table 7: Classification accuracy for each classifier on Set No 2 with static parameters and CLAHE pre-processing.

Pre-trained model	Set No 3 Prediction (%)	CLAHE Prediction (%)	Difference (%)
CNN	25.45	26.36	3.58
VGG16	49.09	51.52	4.95
EFIB0	87.88	87.27	-0.69
INCV3	89.7	83.03	-7.44
MNV2	90.91	89.09	-2.0
RES50	93.94	89.09	-5.16
ENSBL	92.73	92.12	-0.66

Table 8: Classification accuracy for each classifier on Set No 3 with static parameters and CLAHE pre-processing.

Pre-trained model	Set No 4 Prediction (%)	CLAHE Prediction (%)	Difference (%)
CNN	36.26	31.87	-12.11
VGG16	40.66	45.6	12.15
EFIB0	60.44	64.84	7.28
INCV3	62.64	60.44	-3.51
MNV2	65.93	76.92	16.67
RES50	72.89	74.73	2.52
ENSBL	67.03	62.64	-6.55

Table 9: Classification accuracy of all classifiers on Set No 4 with static parameters and CLAHE pre-processing.

4.8 Results Hybrid

Pre-trained model	Set No 1 Prediction (%)	Hybrid-model Prediction (%)	Difference (%)
CNN	46.15	47.25	2.38
VGG16	65.93	73.63	11.68
INCV3	70.33	65.93	-6.26
EFIB0	65.93	75.82	15.0
MNV2	76.92	73.63	-4.28
RES50	76.92	76.92	0.0
ENSBL	74.74	84.62	13.22

Table 10: Classification accuracy of all classifiers on Set No 1 with default parameters and HYBRID architecture.

Pre-trained model	Set No 2 Prediction (%)	Hybrid-model Prediction (%)	Difference (%)
CNN	73.63	73.63	0.0
VGG16	61.54	73.63	19.65
EFIB0	74.73	80.22	7.35
INCV3	81.32	82.42	1.35
MNV2	78.02	84.62	8.46
RES50	84.63	85.71	1.28
ENSBL	89.01	90.11	1.24

Table 11: Classification accuracy of all classifiers on Set No 2 with default parameters and HYBRID architecture.

Pre-trained model	Set No 3 Prediction (%)	Hybrid-model Prediction (%)	Difference (%)
CNN	25.45	26.67	4.79
VGG16	49.09	85.45	74.07
EFIB0	87.88	87.88	0.0
INCV3	89.7	88.18	-1.69
MNV2	90.91	90.3	-0.67
RES50	93.94	90.91	-3.23
ENSBL	92.73	96.97	4.57

Table 12: Classification accuracy of all classifiers on Set No 3 with default parameters and HYBRID architecture.

Pre-trained model	Set No 4 Prediction (%)	Hybrid-model Prediction (%)	Difference (%)
CNN	36.26	39.56	9.1
VGG16	40.66	53.85	32.44
EFIB0	60.44	69.23	14.54
INCV3	62.64	64.84	3.51
MNV2	65.93	67.03	1.67
RES50	72.89	72.53	-0.49
ENSBL	67.03	78.02	16.4

Table 13: Classification accuracy of all classifiers on Set No 4 with default parameters and HYBRID architecture.

4.9 Results Augmentation

Pre-trained model	Set No 1 Prediction (%)	Augmentation Prediction (%)	Difference (%)
CNN	46.15	48.35	4.77
VGG16	65.93	67.03	1.67
EFIB0	65.93	73.63	11.68
INCV3	70.33	73.63	4.69
MNV2	76.92	70.33	-8.57
RES50	76.92	68.13	-11.43
ENSBL	74.74	76.92	2.92

Table 14: Classification accuracy for each classifier on Set No 1 with static parameters and data augmentation.

Pre-trained model	Set No 2 Prediction (%)	Augmentation Prediction (%)	Difference (%)
CNN	67.03	73.63	9.85
VGG16	61.54	65.93	7.13
EFIB0	74.73	71.43	-4.42
INCV3	81.32	75.82	-6.76
MNV2	78.02	82.42	5.64
RES50	84.63	80.22	-5.21
ENSBL	89.01	84.62	-4.93

Table 15: Classification accuracy for each classifier on Set No 2 with static parameters and data augmentation.

Pre-trained model	Set No 3 Prediction (%)	Augmentation Prediction (%)	Difference (%)
CNN	25.45	25.45	0.0
VGG16	49.09	66.67	35.81
EFIB0	87.88	87.88	0.0
INCV3	89.7	81.21	-9.46
MNV2	90.91	88.48	-2.67
RES50	93.94	89.09	-5.16
ENSBL	92.73	94.55	1.96

Table 16: Classification accuracy for each classifier on Set No 3 with static parameters and data augmentation.

Pre-trained model	Set No 4 Prediction (%)	Augmentation Prediction (%)	Difference (%)
CNN	36.26	46.15	27.28
VGG16	40.66	61.54	51.35
EFIB0	60.44	51.61	-14.61
INCV3	62.64	61.54	-1.76
MNV2	65.93	67.58	2.5
RES50	72.89	54.95	-24.61
ENSBL	67.03	69.23	3.28

Table 17: Classification accuracy for each classifier on Set No 4 with static parameters and data augmentation.

References

[1] Jindong Wang and Yiqiang Chen. *Pre-Training and Fine-Tuning*. Machine Learning: Foundations, Methodologies, and Applications. 2022.

[2] Behnaz Abdollahi, Naofumi Tomita, and Saeed Hassanpour. *Data Augmentation in Training Deep Learning Models for Medical Image Analysis*, pages 99–118. Springer International Publishing, 2020.

[3] John Smith and Emily Lee. *Ensemble Methods in Machine Learning*. Springer, New York, NY, 2023.

[4] Devvi Sarwinda, Radifa Hilya Paradisa, Alhadi Bustamam, and Pinkie Anggia. Deep learning in image classification using residual network (resnet) variants for detection of colorectal cancer. *Procedia Computer Science*, 179:423–431, 2021.

[5] Haoting Liu, Beibei Yan, Ming Lv, Junlong Wang, Xuefeng Wang, and Wei Wang. *Adaptive CLAHE Image Enhancement Using Imaging Environment Self-perception*, pages 343–350. Springer, 2017.

[6] Fartash Faghri. Training efficiency and robustness in deep learning. *arXiv preprint arXiv:2112.01423*, 2021.

A Deep Learning Model List

ID	Name	Abbreviation
1	Convolutional Neural Network	CNN
2	Visual Geometry Group 16	VGG16
3	MobileNet Version 2	MNV2
4	Residual Networks (Depth 50)	RES50
5	Inception Version 3	INCV3
6	EfficientNet (Model B0)	EFIB0
7	Ensemble Learning Model	ENSBL

B Feature List

ID	Name	Abbreviation
1	Convolutional Neural Network	CNN
2	Standard Deviation	std
3	Skewness	skewness
4	Kurtosis	kurtosis
5	Angular Second Moment Mean	asm_mean
6	Angular Second Moment Range	asm_range
7	Energy Mean	energy_mean
8	Energy Range	energy_range
9	Contrast Mean	contrast_mean
10	Contrast Range	contrast_range
11	Correlation Mean	correlation_mean
12	Correlation Range	correlation_range
13	Dissimilarity Mean	dissimilarity_mean
14	Dissimilarity Range	dissimilarity_range
15	Homogeneity Mean	homogeneity_mean
16	Homogeneity Range	homogeneity_range
17	Short Run Emphasis Mean	sre_mean
18	Short Run Emphasis Range	sre_range
19	Long Run Emphasis Mean	lre_mean
20	Long Run Emphasis Range	lre_range
21	Gray-Level Non-Uniformity Mean	glnu_mean
22	Gray-Level Non-Uniformity Range	glnu_range
23	Run Length Non-Uniformity Mean	rlnu_mean
24	Run Length Non-Uniformity Range	rlnu_range
25	Run Percentage Mean	rpc_mean
26	Run Percentage Range	rpc_range
27	Tamura Coarseness	tmr_coarseness
28	Tamura Contrast	tmr_contrast
29	Tamura Directionality	tmr_directionality
30	Tamura Roughness	tmr_roughness