

UNIVERSITY OF THESSALY
SCHOOL OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

REALIZATION OF LOW COST NEURAL NETWORK BASED INFERENCE MODELS

Diploma Thesis

Gerasimos Agoras

Supervisor: Georgios Karakonstantis

September 2024



UNIVERSITY OF THESSALY
SCHOOL OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

REALIZATION OF LOW COST NEURAL NETWORK BASED INFERENCE MODELS

Diploma Thesis

Gerasimos Agoras

Supervisor: Georgios Karakonstantis

September 2024



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

**ΥΛΟΠΟΙΗΣΗ ΜΟΝΤΕΛΩΝ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ
ΧΑΜΗΛΟΥ ΚΟΣΤΟΥΣ**

Διπλωματική Εργασία

Γεράσιμος Αγοράς

Επιβλέπων/πουσα: Γεώργιος Καρακωνσταντής

Σεπτέμβριος 2024

Approved by the Examination Committee:

Supervisor **Georgios Karakonstantis**

Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Georgios Stamoulis**

Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Nestor Eumorfopoulos**

Professor, Department of Electrical and Computer Engineering, University of Thessaly

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Gerasimos Agoras

Diploma Thesis

REALIZATION OF LOW COST NEURAL NETWORK BASED INFERENCE MODELS

Gerasimos Agoras

Abstract

Early diagnosis of cancer is pivotal for improving patient outcomes and survival rates. This thesis presents a comprehensive analysis of various machine learning models applied to the early detection of cancer. We take a look on the performance of multiple algorithms that have become the current standard on handling classification problems, using diverse datasets that represent brain tumor and cervical cancer samples. The study emphasizes the importance of feature selection, data pre-processing, and model optimization in achieving high diagnostic accuracy and the impact they have in terms of architectural cost. Our results demonstrate that convolutional neural networks lack excessively in terms of precision but manage to utilize less resources on operation. Additionally, we explore the impact of integrating clinical and genomic data, revealing that combined data approaches significantly enhance model performance. The findings underscore the potential of machine learning as a tool for clinicians in the early diagnosis of cancer, paving the way for more personalized and timely interventions. Future work will focus on utilizing every type of data collected from each case and the development of interpretable models to facilitate clinical adoption.

Keywords:

Early diagnosis, Cancer detection, Machine learning, Support vector machines, Neural networks, Genomic data, High accuracy, Low cost, Image segmentation

Διπλωματική Εργασία

ΥΛΟΠΟΙΗΣΗ ΜΟΝΤΕΛΩΝ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ ΧΑΜΗΛΟΥ ΚΟΣΤΟΥΣ

Γεράσιμος Αγοράς

Περίληψη

Η έγκαιρη διάγνωση του καρκίνου είναι κρίσιμη για τη βελτίωση των αποτελεσμάτων των ασθενών και των ποσοστών επιβίωσης. Η παρούσα διπλωματική εργασία παρουσιάζει μια ολοκληρωμένη ανάλυση διαφόρων μοντέλων μηχανικής μάθησης που εφαρμόζονται στην έγκαιρη ανίχνευση του καρκίνου. Εξετάζουμε την απόδοση πολλών αλγορίθμων που έχουν καθιερωθεί ως το τρέχον πρότυπο για την αντιμετώπιση προβλημάτων ταξινόμησης, χρησιμοποιώντας διαφορετικά σύνολα δεδομένων που αντιπροσωπεύουν δείγματα από όγκους εγκεφάλου και καρκίνο του τραχήλου της μήτρας. Η μελέτη δίνει έμφαση στη σημασία της επιλογής χαρακτηριστικών, της προεπεξεργασίας δεδομένων και της βελτιστοποίησης των μοντέλων για την επίτευξη υψηλής διαγνωστικής ακρίβειας, καθώς και στην επίδραση που έχουν αυτά στο αρχιτεκτονικό κόστος. Τα αποτελέσματά μας δείχνουν ότι τα συνελκτικά νευρωνικά δίκτυα υστερούν σημαντικά όσον αφορά την ακρίβεια, αλλά καταφέρνουν να χρησιμοποιούν λιγότερους πόρους κατά τη λειτουργία. Επιπλέον, εξετάζουμε την επίδραση της ενσωμάτωσης κλινικών και γονιδιωματικών δεδομένων, αποκαλύπτοντας ότι οι συνδυασμένες προσεγγίσεις δεδομένων βελτιώνουν σημαντικά την απόδοση των μοντέλων. Τα ευρήματα αναδεικνύουν τις δυνατότητες της μηχανικής μάθησης ως εργαλείο για τους κλινικούς γιατρούς στην έγκαιρη διάγνωση του καρκίνου, ανοίγοντας τον δρόμο για πιο εξατομικευμένες και έγκαιρες παρεμβάσεις. Μελλοντικές εργασίες θα επικεντρωθούν στη χρήση κάθε τύπου δεδομένων που συλλέγονται από κάθε περίπτωση και στην ανάπτυξη ερμηνεύσιμων μοντέλων για τη διευκόλυνση της κλινικής υιοθέτησης.

Λέξεις-κλειδιά:

Έγκαιρη διάγνωση, Ανίχνευση καρκίνου, Μηχανική μάθηση, Μηχανές υποστήριξης διανυσμάτων, Νευρωνικά δίκτυα, Γονιδιωματικά δεδομένα, Υψηλή ακρίβεια, Χαμηλό κόστος, Τμηματοποίηση εικόνας

Table of contents

Abstract	x
Περίληψη	xi
Table of contents	xiii
Abbreviations	xv
1 Introduction	1
1.1 Topic of thesis	1
1.1.1 Contribution	2
1.2 Structure of the thesis	2
2 Background	3
2.1 Metrics	3
2.2 Knowledge	5
3 Brain tumor	9
3.1 SOM-PSVM	9
3.2 3D MRI auto	12
3.3 IJAYA-ELM	18
3.4 DWA-DNN	25
3.5 CNN-SVM TH	31
3.6 CNN-NADE	35
3.7 CNN T1-CE	37
3.8 DNN-TL	41
3.9 ACV-DHOA	45

3.10 Overall comparison	52
3.10.1 Accuracy comparison	52
3.10.2 Cost comparison	53
4 Cervical cancer	57
4.1 SVM Cerv	57
4.2 CNN-superpixel	61
4.3 DeepPap	65
4.4 AlexNet-SVM	70
4.5 SSAE-SC	72
4.6 CNN-SVM	77
4.7 DeepCervix	84
4.8 MFFOA-DL3	88
4.9 Overall comparison	92
4.9.1 Accuracy comparison	92
4.9.2 Cost comparison	93
5 Conclusion	95
5.1 Overview and Conclusion	95
5.2 Future work	96
Bibliography	97

Abbreviations

ML	Machine Learning
DL	Deep Learning
NN	Neural Network
CNN	Convolutional Neural Network
ANN	Artificial Neural Network
DNN	Deep Neural Network
KNN	Kohonen Neural Network
Acc	Accuracy
Prec	Precision
Rec	Recall
F-Meas	F-Measure
H-mean	Harmonic mean
ASM	Angular Second Moment
HOM	Homogeneity
IDM	Inverse Difference Moment
EN	Entropy
VAR	Variance
MRI	Magnetic resonance imaging
SVM	Support Vector Machine
DT	Decision Tree
RF	Random Forest
GLCM	Gray-level Co-occurrence Matrix
SOM	Self-organizing Mapping
SOM-PSVM	SOM clustering with Proximal SVM
3D MRI auto	3D MRI based automated model

IJAYA-ELM	Optimized Jaya algorithm, improved ELM and orthogonal ripplelet-II transform
DWA-DNN	Deep wavelet autoencoder-based DNN
CNN-SVM TH	Hybrid CNN-SVM Threshold Segmentation
CNN-NADE	DNN hybrid model using CNN and NADE
CNN T1-CE	T1-weighted CE-MRI CNN model
DNN-TL	DNN-based MRI classification model with Transfer Learning approach
ACV-DHOA	Adaptive Coefficient Vector-based Deer Hunting Optimization Algorithm
SVM Cerv	SVM-based Cervical cancer detection
CNN-superpixel	CNN-based superpixel segmentation of cervical cancer cells
DeepPap	CNN-based deep feature extraction
AlexNet-SVM	SVM classifier with pre-trained AlexNet on cervical cancer
SSAE	Stacked Sparse Autoencoder
SSAE-SC	Two-level classification of abnormalities, SSAE to Softmax Classifier
MFFOA-DL3	Modified Firefly Optimization Algorithm with DL enabled

Chapter 1

Introduction

The advent of ML and artificial intelligence markedly enhances efforts in cancer prevention and treatment. Artificial intelligence, often characterized as a collection of computer-coded programs or algorithms, utilizes sophisticated data analysis and pre-established instructions to forecast and decide on diverse facets of a disease. ML, a quantum of artificial intelligence, encompasses a cluster of algorithms engineered to autonomously acquire knowledge and enhance functionality through experience. In essence, ML represents a subset of AI dedicated to crafting algorithms capable of assimilating data and iteratively refining their efficacy over time.

1.1 Topic of thesis

In this thesis, we explore the realization of the existing inference model and their computational depth, aiming to bridge the gap between the effectiveness of NNs and the practical constraints of deployment in resource-constrained environments. With the help of various sources, we investigate optimization techniques and hardware-efficient implementations to mitigate the computational overhead associated with NNs. By leveraging advancements in training and classification algorithm, research teams from around the world aim to reduce the memory footprint and computational complexity of NN models without compromising their predictive accuracy. We discuss the implications of these considerations for real-world applications, emphasizing the importance of balancing model performance with cost-effectiveness in deployment scenarios.

1.1.1 Contribution

Through this study, we seek to provide insights into the development of practical and scalable solutions for deploying NN-based inference models in resource-constrained environments. By addressing the challenges of computational cost and efficiency, we aim to enable the widespread adoption of NNs across diverse applications, ultimately advancing the state-of-the-art in ML deployment. One field that can take a significant leap of such advancements is the prediction of cancer and it is the one that we will focus on for the rest of this thesis.

Our contribution of this thesis is:

1. Study of inference models on brain tumor and cervical cancer
2. Statistical analysis of the performance achieved from the models
3. Focus on the cost of the methods and what makes an algorithm low-cost

1.2 Structure of the thesis

On the sections that follow, first we will take a look on the required background to help with the information that will follow. Next chapter focuses on brain tumor methods that integrate early diagnosis by using various algorithms, comparing them both in terms of accuracy and architectural depth and referencing the key points of interest. The following chapter has a similar structure but this time we focus on cervical cancer methods. Lastly, we focus on how the knowledge we already have on these fields could push future researchers both on the same cancer types and other deadly diseases. Also, we state what other types of data processing there are other than image segmentation.

Chapter 2

Background

Before diving into the essence of this thesis, there's a number of information we need to refer to so that the analysis and the terms used are clear.

2.1 Metrics

A number of different methods are evaluated with similar metrics, so let's take a look at those.

- TP (true positive), correctly classified malignant sample.
- TN (true negative), correctly classified clean sample.
- FP (false positive), incorrectly diagnosed clean sample as detected malicious.
- FN (false negative), incorrectly diagnosed malignant sample as clean.
- Accuracy, it gives an overall measure of how often a prediction is correct.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision, it measures the proportion of TP instances out of all instances that were predicted as positive.

$$Prec = \frac{TP}{TP + FP}$$

- Recall, it tells you how many of the actual positive instances were captured by the model.

$$Rec = \frac{TP}{TP + FN}$$

- F-measure or F1-score, is a metric that combines both Prec and Rec into a single score, providing a balanced assessment of a model's performance, particularly in cases where there is an uneven trade-off between Prec and Rec.

$$F - Measure = \frac{2 \times Prec \times Rec}{Prec + Rec}$$

- Harmonic mean, is a type of average that is particularly useful for combining rates or ratios. It is less sensitive to large values and gives more weight to smaller values.

$$H - mean = 2 \times \frac{Sens \times Spec}{Sens + Spec}$$

- Angular Second Moment / Energy, is the sum of the squares of the elements. $P(i,j)$ depicts the pixel value of the i -th row and j -th column.

$$ASM = \sum_{i=1}^N \sum_{j=1}^N P(i,j)^2$$

- Homogeneity, it measures how close the distribution of elements in the GLCM is to the GLCM diagonal. Quoting MathWorks, developer company of MATLAB, "GLCM is a statistical method of examining texture that considers the spatial relationship of pixels".

$$HOM = \sum_{i=1}^N \sum_{j=1}^N j = 1^N \frac{P(i,j)}{1 + |i - j|}$$

- Inverse Different Moment, a texture measure derived from the GLCM that quantifies the local homogeneity of an image.

$$IDM = \sum_{i=1}^N \sum_{j=1}^N j = 1^N \frac{P(i,j)}{1 + (i - j)^2}$$

- Entropy, a measure of the randomness or complexity of an image's texture.

$$EN = - \sum_{i=1}^N \sum_{j=1}^N j = 1^N P(i,j) \log(P(i,j))$$

- Variance, measure of the spread or dispersion of pixel intensity values within an image region.

$$VAR = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2$$

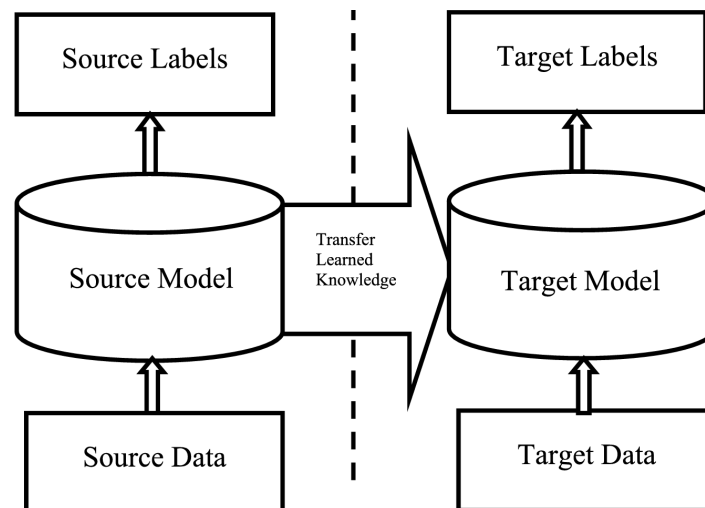


Figure 2.1: The transfer learning approach.

2.2 Knowledge

Since we are going to take a look on various types of NNs, it would be important to state the main structures from which the models are built:

Convolutional neural networks are known for their effectivity with image, speech, or audio signal inputs. They have three main types of layers, which are:

- Convolutional layer, which apply filters that through computations construct the feature vectors.
- Pooling layer, which is responsible for reducing the spatial dimensions of the input, controlling overfitting and extracting dominant features.
- Fully-connected (FC) layer which through its complex structure performs high-level decision making for the features that will be extracted.

In the other hand, there also exist deep neural networks which compared to CNNs, there are two main differences. First, a DNN can include multiple hidden layers that gives the network its "deep" structure. Due to its complicated nature, it can handle any type of data (structured, unstructured), but is also less optimized for image-related tasks. Also, let's give a quick introduction to some of the algorithms and methods we are going to see in the following sections: SVM is a supervised ML algorithm used for classification and regression. It works by putting the input elements in a feature space of higher dimensionality in order to separate them into different classes. They are commonly used in classification problems. In our problem, it takes

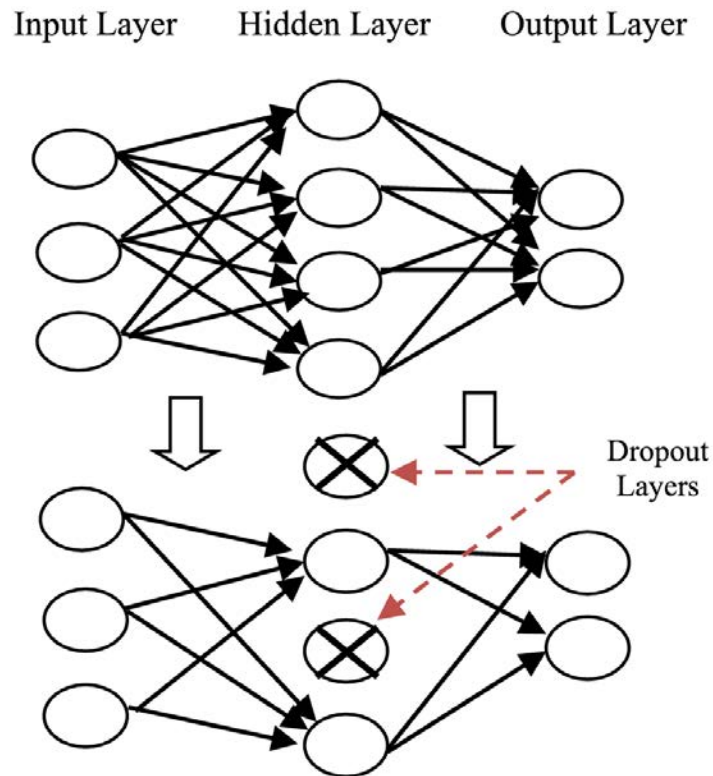


Figure 2.2: Dropout layer example with 50% dropout probability.

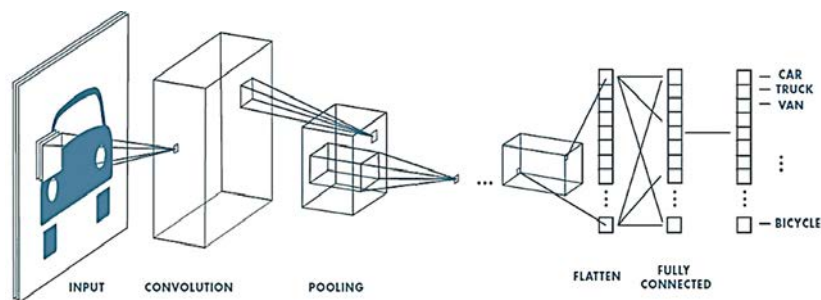


Figure 2.3: CNN general architecture of image analysis.

responsibility of separating the harmful tumors from those that are benign.

DTs are structures that follow the well-known tree formation found on several data structures of computing. The root node of the tree represents the initial decision made, The internal nodes are the possible decisions made after that with the branches showing the possible outcomes of each decision. The leaf nodes represent the DT outcome. Traversal of DTs is one of their advantages, as there are multiple ways to approach it, depending on the approach of the developer.

The RF algorithm is an ensemble learning method used for classification and regression tasks. It builds multiple decision trees, each trained on a random subset of the data and features,

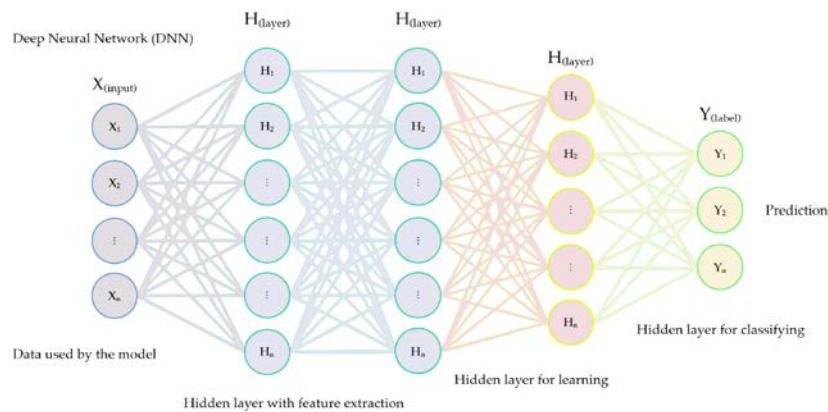


Figure 2.4: DNN general architecture. This is a simple example of how a DNN may be built.

and combines their outputs to make predictions. For classification, it uses majority voting across the trees, and for regression, it averages their predictions. Random Forest is robust to overfitting, handles high-dimensional data well, and provides strong predictive performance, though it can be less interpretable and slower than simpler models. Its key strengths include randomness in data and feature selection, which helps improve accuracy and reduce bias.

Chapter 3

Brain tumor

Brain tumor is one of the dreadful cancers we deal with. It is developed when the cells inside the skull grow abnormally. MRI (Magnetic Resonance Imaging) is the most common test to detect brain tumor. It is a non-invasive imaging technique that uses magnets, radio waves, and a computer to produce detailed images of the body's internal structures, especially soft tissues. It doesn't use radiation, making it safer for repeated use. MRI is commonly used to image the brain, joints, heart, and other organs. There are different types of brain tumors like 'Glioma' tumor, 'Meningioma' tumor and 'Pituitary' tumor.

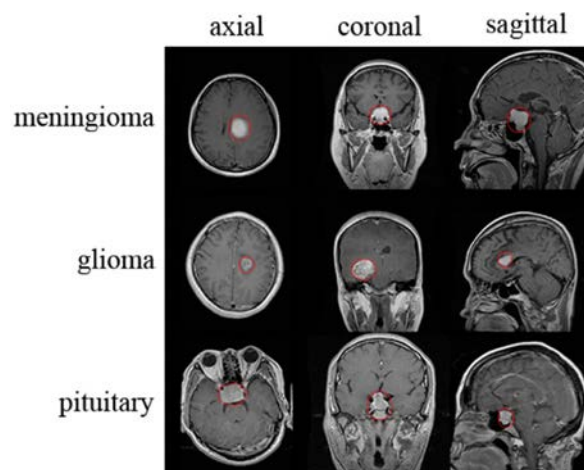


Figure 3.1: Three types of brain tumor.

3.1 SOM-PSVM

In 2015, Vainshnavee and Amshakala [1] used Proximal SVM and SOM, an ANN used for unsupervised learning of MRI brain image segmentation. Two different approaches are

studied in this paper, one that is fast and efficient histogram-based and one based on the inherent extracted image features, a robust architecture that handles conditions like noise and bad image intensity. SOMs address several critical aspects of the problem:

- MRI scans are often high-dimensional. SOMs help in reducing the dimensionality of this data while preserving the topological properties, making it easier to visualize and analyze complex structures like tumors.
- SOMs can effectively cluster similar patterns in the imaging data. By learning the distribution of these patterns, SOMs can help in segmenting and identifying regions that are likely to be tumors.
- SOMs do not require labeled data for training, which makes them particularly useful in scenarios where labeled datasets are scarce or expensive to obtain.
- SOMs can be used to extract meaningful features from the raw imaging data. These features can then be used for further classification or segmentation tasks, improving the Acc and robustness of the segmentation.
- SOMs provide a low-dimensional representation of the high-dimensional data, which can be visualized in a 2D map.
- Medical images often contain noise and artifacts. SOMs can help in smoothing and de-noising the data by grouping similar patterns together, leading to more accurate segmentation results.

The data processed on this research come from the Internet Brain Segmentation Repository (IBSR), Massachusetts General Hospital (1.0 has 20 T_1 -weighted volumetric images, 2.0 has 18 T_1 -weighted processed by Center for Morphometric Analysis (CMA)) and from the "Virgen de las Nieves" hospital (Granada, Spain) (VNH).

The goal set by the pre-processing of the resources is the clean-up of the input's noise. Some implementations that have been proposed in the past are Brain Surface Extractor (BSE), Brain Extraction tool (BET), Minneapolis Consensus Strip (McStrip) and Watershed Algorithm (HWA). For ISBR 1.0, all these work but for the second version, BET is used to remove the scalp from the image. The pattern detection is done in binary coding, 1 depicting high value pixels and 0 the background that are classified by a threshold value.

To achieve correct classification, first is collecting the data from the volume image histogram to build feature vectors by applying histogram equalization. It achieves that by assigning each pixel to an intensity value to improve contrast of the image. This is advantageous, as every aspect known about the area, such as texture and shape and mark different metrics such as mean intensity, number of occurrences and variance values.

Then, we utilize GLCM in order to extract the texture features of the different image parts. As they are grayscaled from the binary classification, the co-occurrence matrix of the image is needed. The features extracted gives the property of the texture, so the elements get stored in a knowledge-based fashion. That distinguishes texture that signals tumor to normal surface. The features that are detected and values of recognisability are as follow:

- Contrast
- ASM, the sum of the squares of the elements in the GLCM.
- HOM, it measures how close the distribution of elements in the GLCM is to the GLCM diagonal.
- IDM, a texture measure derived from the GLCM that quantifies the local HOM of an image.
- EN, a measure of the randomness or complexity of an image's texture, derived from the GLCM.
- VAR, a measure of the spread or dispersion of pixel intensity values within an image region.

The evaluation is done on MATLAB. As we see on the table of results (fig.3.2), the Acc starts on 89% on the simplest variant, on the model accommodated only with SVM and SOM. When the Hierarchy Feature Selection is added, the Acc improves by 3%, as the data handled are sorted and evaluated better. Adding the GLCM with the reduction of features from PCA, the Acc improves even further, up to 94%. Finally, switching from SVM to PSVM pushes the performance of the model to a competitive 97%, an outcome that puts it right along other interpretations taken from other researchers that we discuss on this thesis. Testing the model on a more resent and complicated dataset as the BRATS 2015 dataset, the results recorded were an Acc of 92%, Rec of 94% and Prec of 93%.

Techniques	Accuracy%
SVM-SOM	89
SVM- HFS-SOM	92
SVM- HFS-SOM-GLCM-PCA	94
PSVM- HFS-SOM-GLCM-PCA	97

Figure 3.2: Accuracy comparison as we add optimizations to the SOM-PSVM model.

3.2 3D MRI auto

In 2017, Abbasi and Tajeripour [2] suggested their own model for brain tumor detection by introducing a 3D automated I/O model that follows a similar path to the previous implementation, as it is based on segmentation, histogram pattern detection and, in general, image processing.

MRI is highly rated by the medical field as one of the most common and successful imaging technique, as it can give critical information to the doctors about the soft tissue of the human body anatomy, making the diagnostic check more accurate on brain scanning. Multiple MRI methods are being used on patients as to lessen the risk of a false prediction of the tumor. Some of these are T_1 weighted MRI, T_i weighted MRI with contrast enhancement (T_{1c}) where the tumor borders are indicated by brighter spots as a result of the contrast compared to brain vessels, T_2 weighted MRI and T_2 with fluid-attenuated inversion recovery (T_{2Flair}) which makes easier to recognise edema from cerebrospinal fluid (CSF). On both T_2 variants, the Edema area surrounding the tumor looks bright. An example of each case can be seen in fig.3.3.

Automation in a field like this could be described as more than effective, closer to desired. This can not be easy due to the complexity the texture analysis, as the tumor can differ on shape, create irregularity, discountinuity and face problems with detecting where gliomas start and end. These setbacks, although, cannot push back a breakthrough on a field like that as the need for a solution is vital, as brain tumors is one of the most common mortality issues considered as an essential priority for health care societies. An overview of the implementation that we are discussing can be seen in fig.3.4.

On the pre-processing step of the MRI images priority is the reduction the noise introduced either from external factors. The success rate of clearing the input is measured through the decrease of the signal-to-noise ratio. The proposed method was introduced by Tustison et

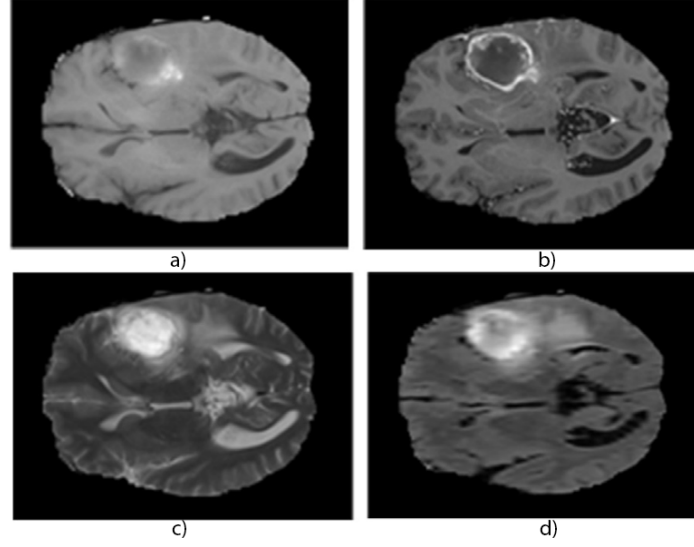


Figure 3.3: MR scanning modalities: a) T_1 , b) T_{1c} , c) T_2 , d) T_{2Flair}

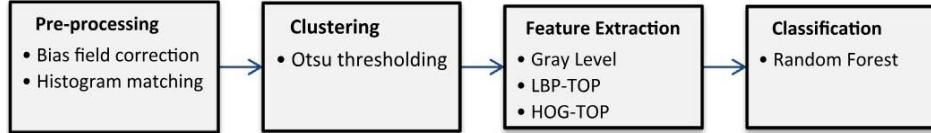


Figure 3.4: Illustration of the pipeline followed on the 3D MRI auto model. Block illustration taken from [2].

al.[3] and it applies correction on bias parameters estimated via Legendre polynomials on MRI images using the N4ITK method. It achieves minimization of the total energy which then is normalized by a histogram-matching algorithm implemented in ITK [4].

After that, unsupervised Otsu algorithm separates the high interest region from the rest of the Flair image. This clustering algorithm achieves tuning of the image, as it automatically sets the threshold of the image.

Let's consider an image with L grey level ($0, \dots, L - 1$) and f_i is the number of pixels with grey level of i ; This means that in total with have pixels $N = (f_0 + f_1 + \dots + f_{L-1})$.

The grey-level probability of an image is equal to:

$$p_i = \frac{f_i}{N}, p_i \geq 0, \sum_{i=0}^{L-1} p_i = 1$$

If clustering creates k segments, $k - 1$ thresholds are needed to be put. The cumulative probability w_k and mean grey-level μ_k for each cluster c_k are as follows:

$$w_k = \sum_{i \in c_k} p_i, \mu_k = \sum_{i \in c_k} i \cdot \frac{p_i}{w_k} \quad k \in \{0, 1, \dots, K - 1\}$$

The mean intensity of the whole MRI image μ_T and between class variance σ_B^2 are computed as follows:

$$\mu_T = \sum_{i=0}^{L-1} i \cdot p_i = \sum_{k=0}^{K-1} \mu_k \cdot w_k, \sigma_B^2 = \sum_{k=0}^{K-1} w_k \cdot (\mu_k - \mu_T)^2$$

The optimum Thresholds $(t_0^*, t_1^*, \dots, t_{k-2}^*)$ can be determined by maximizing the between class variance as:

$$t_0^*, t_1^*, \dots, t_{k-2}^* = \operatorname{argmax}\{\sigma_B^2(t_0, t_1, \dots, t_{k-2})\}$$

After follows the feature extraction from the samples for the classification of the pixels as grey material (GM), white material (WM) and Cerebro Spinal Fluid (CSF). An easy indicator of which region fits on each bucket is through the intensity of the image. Unfortunately, the radio signals produced from the frequencies in which MRI images are made in affect the results with noise and heterogeneity which means the results are not reliable for intensity analysis. The factor that seems most reliable is texture, as different regions of the tumor are indicated via the textural patterns on the MRI result. A simple alternative of a pattern matching method is local binary pattern (LBP-TOP) which compares the pixels' intensity of a circular neighbourhood of the image to the center point of it that results in a binary classification value for each pixel:

$$LBP_{P,R} = \sum_{i=0}^{P-1} s(g_i - g_c) 2^i s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$

where P represents the number of neighbouring pixels, R the neighbourhood radius, g_i is the intensity of each p , g_c is the intensity of central p .

Uniformity in bit patterns, particularly in the context of Local Binary Patterns (LBP), refers to a specific property of binary sequences used for texture analysis in image processing. In LBP, uniform patterns are defined as binary sequences that contain at most two 0-1 or 1-0 transitions when viewed as a circular sequence. This concept helps in simplifying the representation and improving the robustness of texture features. Labels are given to the neighbourhood from the aforementioned area, which show the best matching points. We see an example of this in fig.3.5.

Each texture value can be described in three dimension: **XY-LBP**, **XY-LBP**, **YT-LBP**. X and Y represent the spatial coordinates while T is time. A LBP-voxel is on the crossroad of the three planes of XY , XT and YT with eight neighbor delineated for each. The final

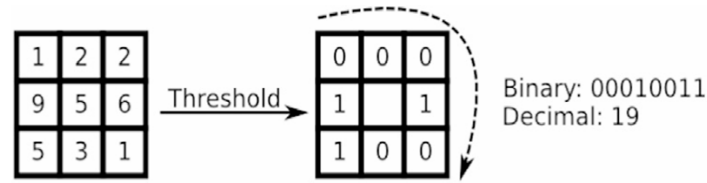


Figure 3.5: Binary code of each pixel in local binary patterns. It operates by analyzing the relationship between a pixel and its neighboring pixels within a defined neighborhood.

feature vector is achieved by positioning the values of the local binary pattern. To put that into perspective, we can see a neighborhood point of each voxel in three orthogonal planes on fig.3.6.

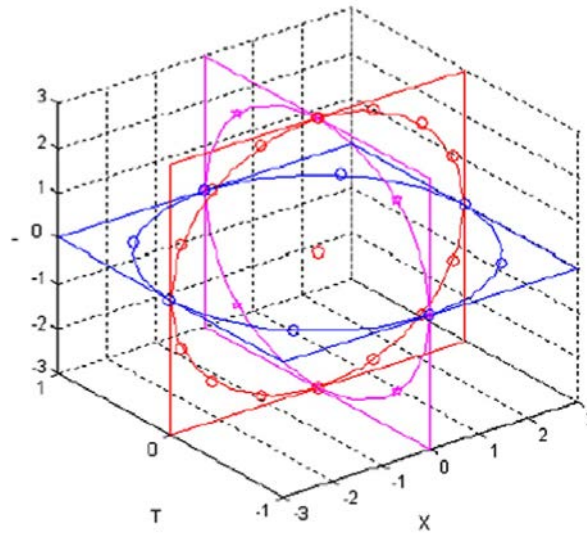


Figure 3.6: Voxel representation on three orthogonal planes of (X,Y) and T (time). Example taken from [2].

On this implementation, LBP-TOP is used as an assisting method, in order to extend HOG to 3D images. HOG or Histogram orientation gradient is a computer vision, object detection method that divides the image in small areas or cells. First is the filtering via Sobel kernel to get the image coordinates' gradients.

The magnitude and angle of the image gradient are computed as:

$$|G(i, j)| = \sqrt{(G_x(i, j))^2 + (G_y(i, j))^2}$$

$$\theta_G(i, j) = \tan^{-1}\left(\frac{G_y(i, j)}{G_x(i, j)}\right)$$

The gradient histogram is calculated by dividing the angle θ_G into n equal distances, where n is the number of gradient directions. An example of that is shown in fig.3.7. The value

calculated is stored in the central pixel. Dalal and Triggs [5] tried a 9-dimensional histogram, which means that each pixel has a vector of length equal to 9.

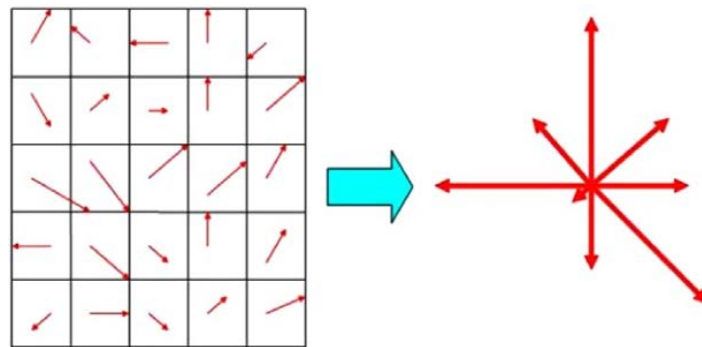


Figure 3.7: An example of a cell and histogram orientation gradient. Example taken from [2].

The learning algorithm chosen by the researchers is the RF classification algorithm which achieves classification and regression by constructing a number of decision trees at training time and producing the mean prediction of the individual trees. The ability to operate with a lot of different inputs increases the classification Acc significantly.

Moving to the evaluation of the implementation, they used the Brats2013 dataset with 1 mm isotropic resolution. The dataset has 30 real glioma patients (20 high grade (HG) and 10 low grade (LG)), and 50 simulated glioma patients (25 of each grade). The classes that have been set manually are: **1. necrosis**, **2. edema**, **3. non-enhancing tumor**, **4. enhancing tumor** and **0. everything else**.

The categories that tumor are set in are:

- Complete tumor (necrosis, edema, non-enhance tumor, enhance tumor)
- Tumor core (non-enhance tumor, enhance tumor)
- Enhance Tumor (enhance tumor)

The region of interest is only 20% of the image which helps lessen the time needed for evaluation. Pre-processing the image into areas speeds up the method. Feature extraction takes, on average, **1516** seconds per image. This process takes only $\frac{1}{4}$ **of the starting time** when clustering leads before feature extraction.

The evaluation step is done through ten-fold cross validation for real high grade glioma, leave one-out scheme for real low grade glioma and two-fold cross validation for synthetic data. Two coefficients are used to indicate the Acc, the Dice and the Jaccard. The equations that

describe them are:

$$Dice(A, B) = \frac{2|A \cap B|}{|A| + |B|}, Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Dice coefficient depicts the overlap between manual segmentation and automatic segmentation. Jaccard coefficient depicts their similarity. Both have values between 0 and 1. The measurements for comparison of the results are mean and standard deviation $3.8(\alpha')$ $3.8(\beta')$ $3.8(\delta')$. What we understand from this outcome is that there's better performance both in terms of mean value and less mean deviation compared to other implementation published until then. The authors also compared their architecture to BraTumIA (Brain Tumor Image Analysis), an application that does 3D image analysis on brain tumors, used by oncologists. As inputs, it gets four different MRI sequences (T_1 , T_{1c} , T_2 , T_{2Flair}) and as output we get data about the volume of the tumor and its sub-parts that we mentioned before. As we see on the appropriate table (3.8(γ')), the results taken from the proposed new method seems to handle better the data that we have on both grades of glioma, operating around 93% Acc while Cordier's software manages to get performance about 83% accurate.

Comparison results in real HG.

Approach	Dice Complete tumor	Dice Tumor core	Dice Enhancing tumor	Jaccard Complete tumor	Jaccard Tumor core	Jaccard Enhancing tumor
Proposed method	0.8377 ± 0.105	0.759 ± 0.15	0.761 ± 0.19	0.733 ± 0.14	0.635 ± 0.19	0.647 ± 0.21
Meier et al.	0.802 ± 0.124	0.691 ± 0.220	0.698 ± 0.247	0.684 ± 0.153	0.561 ± 0.212	0.578 ± 0.232
Cordier et al.	0.79 ± 0.17	0.60 ± 0.26	0.59 ± 0.25	–	–	–
Festa et al.	0.83 ± 0.08	0.7 ± 0.24	0.75 ± 0.16	0.72 ± 0.12	0.58 ± 0.25	0.63 ± 0.18

Comparison results in real LG.

Approach	Dice Complete tumor	Dice Tumor core	Dice Enhancing tumor	Jaccard Complete tumor	Jaccard Tumor core	Jaccard Enhancing tumor
Proposed method	0.792 ± 0.11	0.652 ± 0.08	0.44 ± 0.26	0.66 ± 0.14	0.503 ± 0.12	0.33 ± 0.24
Meier et al.	0.764 ± 0.105	0.585 ± 0.229	0.201 ± 0.326	0.628 ± 0.121	0.446 ± 0.228	0.152 ± 0.150
Nicolas et al.	0.76 ± 0.18	0.64 ± 0.21	0.44 ± 0.4	–	–	–
Festa et al.	0.72 ± 0.17	0.47 ± 0.23	0.21 ± 0.35	0.58 ± 0.19	0.33 ± 0.19	0.17 ± 0.28

Comparison results with BraTumIA software in real HG.

Approach	Dice Complete tumor	Dice Tumor core	Dice Enhancing tumor	Jaccard Complete tumor	Jaccard Tumor core	Jaccard Enhancing tumor
Proposed method	0.8377 ± 0.105	0.759 ± 0.15	0.761 ± 0.19	0.733 ± 0.14	0.635 ± 0.19	0.647 ± 0.21
BraTumIA software	0.8178 ± 0.068	0.692 ± 0.195	0.709 ± 0.21	0.696 ± 0.094	0.554 ± 0.18	0.58 ± 0.20

Comparison results in synthetic data sets of HG and LG.

Approach	Dice Complete Tumor (HG)	Jaccard Complete Tumor (HG)	Dice Complete Tumor (LG)	Jaccard Complete Tumor (LG)
Proposed method	0.93 ± 0.04	0.87 ± 0.06	0.90 ± 0.02	0.82 ± 0.04
Cordier et al.	0.84 ± 0.08	–	0.83 ± 0.4	–

Figure 3.8: Comparison Results taken in 3D MRI auto in real HG and real LG.

3.3 IJAYA-ELM

In order to improve MRI images (inputs) as per quality and, thus, effectiveness, Nayak et al. [6] developed a system that introduces Orthogonal Discrete Ripplet-II Transform (O-DR2T). That was implemented as the regular Fourier transformation is less suitable for image feature recognition, as it cannot handle point singularities while wavelet transform is not suited for 2D singularities. As we can understand, the problem that faced is scalability, as there was not a method to cover both types of singularities mentioned, thus the Ripplet-I transform was proposed, which generalizes the curvelet transform by adding two parameters: c for support and d for degree. If c and d are set to 1 and 2 accordingly, Ripplet-I transform becomes curvelet transform. Ripplet-II is the improved version that they proposed for faster coefficient convergence and giving sparse images more edges. To put the Ripplet-II transform in math, let's say that we have a 2-variable function $g(x, y)$, we can describe the transform formula in polar coordinates as:

$$RT2_g(s, t, d, \theta) = \int \int (\bar{\psi}_{s,t,d,\theta}(\rho, \alpha) g(\rho, \alpha) \rho) d\rho d\alpha$$

where $g(\rho, \alpha)$ is the polar coordinates of $g(x, y)$, $\psi_{s,t,d,\theta} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ the Ripplet-II function and $\bar{\psi}$ the complex conjugate of ψ .

$$\psi_{s,t,d,\theta}(\rho, \alpha) = s^{-1/2} \phi((\rho \cos^d((\theta - \alpha)/d) - t)/s)$$

$\phi : \mathbb{R} \rightarrow \mathbb{R}$ is a smooth univariate wavelet function and $s > 0$, $t \in \mathbb{R}$, $d \in \mathbb{N}$ and $\theta \in [0, 2\pi)$ indicate scale, translation, degree and orientation parameters, respectively. By tuning these parameters, Ripplet-II transform can capture structural information along arbitrary curves, using what we know, we describe the Ripplet-II formula as:

$$RT2_g(s, t, d, \theta) = \langle \phi_{s,t}(r), GR_d[g] \rangle$$

where $GR_d[g]$ is the GRT of function g and is defined as:

$$GR_d(r, \theta) = \int \int g(\rho, \alpha) \delta(r - \rho \cos^d((\alpha - \theta)/d)) \rho d\rho d\alpha$$

The discrete version of Ripplet-II transform (DR2T) can be defined as:

$$g(\rho, \alpha) \xrightarrow{\text{DGRT}} GR_d[g](\rho, \theta) \xrightarrow{\text{1D-DWT}} RT2_g(s, t, d, \theta)$$

in which the discrete GRT (DGRT) of g is first computed and subsequently, the 1D discrete WT (DWT) of the DGRT of g is computed. The computing procedure for discrete Ripplet-II

transform becomes more simpler when $d = 2$. In this case, the GRT is dubbed as 'parabolic Radon transform and is defined as:

$$GR_2(\rho, \theta) = 2\sqrt{r}R[g(\rho'^2, 2\alpha')](\sqrt{r}, \theta/2)$$

where $R[g(\rho, \alpha)](r, \theta)$ is the classical Radon transform (CRT) in polar coordinates. However, in general, the GRT of the function g for $d > 0$ takes the form in Fourier domain as:

$$GR_d^F(r, \theta) = 2 \sum_{n=-\inf}^{+\inf} \left[\int_r^{\inf} \int g(\rho, \alpha) e^{-in\alpha} d\alpha \times (1 - (r/\rho)^{2/d})^{-1/2} \times T_n d((r/\rho)^{1/d}) d\rho \right] e^{in\theta}$$

where $T_n(\cdot)$ denotes the Chebyshev polynomial of degree n .

To summarize, the forward DR2T with $d = 2$ of an input image can be computed as follows:

1. Convert the input function from Cartesian coordinates to polar coordinates ($g(x, y) \rightarrow g(\rho, \alpha)$). Replace (ρ, α) ($\rho'^2, 2\alpha'$) in (ρ, α) . Subsequently, generate a new image $g'(x, y)$ by interpolation after converting polar coordinates (ρ', α') to Cartesian coordinates (x, y) . The variables x and y hold integer values.
2. Employ discrete CRT on $g'(x, y)$ that produces $R(r', \theta')$ and then substitute (r', θ') with $(\sqrt{r}, \theta/2)$ in $R(r', \theta')$. Then follows the DGRT coefficients $GR_2(r, \theta)$.
3. Apply 1D DWT to DGRT coefficients w.r.t. r and obtain the discrete Ripplet-II coefficients.

In order for this transform to work on the data we process, orthogonal Ripplet-II transform is introduced, which makes the last form of the formula as:

$$RT2_g^{orth}(s, t_1, t_2, d) = 2 \sum_{n=-\inf}^{+\inf} \int \int \frac{1}{s} \bar{\phi}\left(\frac{r-t_1}{s}\right) \bar{\phi}\left(\frac{r-t_2}{s}\right) \int_r^{+\inf} \int g(\rho, \alpha) e^{-in\alpha} d\alpha \times (1 - (r/\rho)^{2/d})^{-1/2} \times T_{nd}((r/\rho)^{1/d}) d\rho e^{in\theta} dr d\theta$$

In order to extract the features of the MR images, O-DR2T is introduced on the training stage and as an outcome we get precise coefficients. From them, we build a coefficient vector of size $(D = m * n)$. After running every training sample, the final feature matrix comes as a result. A more precise representation of the feature extraction algorithm can be seen on the Algorithm 1.

The model also has Contrast Limited Adaptive Histogram Equalization (CLAHE), an advanced form of histogram equalization used in image processing to enhance the contrast of

Algorithm 1 Feature extraction using orthogonal discrete Ripplet-II transform**Input:** N input images: $g[x,y]$; $0 \leq x < m, 0 \leq y < n$ **Output:** Feature matrix: F_M of size $N \times D$

- 1: **for** each image $g[x, y] \in N$ **do**
- 2: Transform $g(x,y)$ into polar coordinates $g(\rho, \alpha)$ and substitute (ρ, α) with $(\rho'^2, 2\alpha')$
- 3: Transform polar coordinates (ρ', α') to Cartesian coordinates (x, y) and obtain another image $g'(x, y)$ by 2D interpolation
- 4: Compute 1D FFT of $g'(x, y)$, $G'(u, v)$ along θ (columns)
- 5: Compute $GR_d(r, \theta)$ in Fourier domain
- 6: Compute inverse 1D FFT g_{inv} on $GR_d(r, \theta)$ along θ (columns)
- 7: Apply 2D DWT on g_{inv} along r and θ , and obtain the D -dimensional ($= m * n$) coefficients
- 8: Arrang the coefficients in a vector of size $1 \times D$ and store in F_M
- 9: **end for**
- 10: Obtain a feature matrix F_M containing all vectors

an image. Unlike standard histogram equalization, which applies a single transformation to the entire image, CLAHE operates on small regions in the image, called tiles or blocks, and performs local contrast enhancement. Additionally, CLAHE limits the amplification of noise by clipping the histogram at a predefined value, which prevents over-enhancement of noise in homogeneous regions.

Unfortunately, since the features generated by O-DR2T increase significantly the computational overhead, measures have to be taken. Thus the research team introduce PCA, a frequently used method that improves the locality of the features by making high dimensional input to lower dimensional ones. Linear discriminant analysis (LDA) serves the purpose of segmentation improvements while also alternating it as it is not built to process high dimensional - small sample size data, as the ones we are working with. The PCA+LDA technique makes sure that singularity isn't obtained on the within-scatter matrix by introducing $D+C$ samples, where D is the dimension of the feature vector and C the number of different classes. The steps of this process are:

1. Input M -dimensional data from the D -dimensional data through PCA
2. Reduce to l -dimensional data using LDA ($l \ll M < D$)

The learning method used is known as IJaya-ELM, which uses the improved Jaya algorithm [7] (fig.3.9) to minimize the norm of the output weights in order to enhance the convergence

of the Extreme Learning Machine (ELM) [8] which avoids the limitations of gradient based learning schemes and can achieve multi-class classification. This is done following the steps below:

1. If s is the solutions vector and j the current iteration, we initialize s with randomly valued input weights of same dimensions and hidden biases within the range of $[-1, 1]$.

$$s_j = [w_{11}^h, w_{12}^h, \dots, w_{1l}^h, w_{21}^h, w_{22}^h, \dots, w_{2l}^h, w_{nh1}^h, w_{nh2}^h, \dots, w_{nhl}^h, b_1, b_2, \dots, b_{nh}]$$

2. The evaluation of the aforementioned values is done using the fitness function.

$$f() = \sqrt{\frac{\sum_{j=1}^{N_\nu} \|\sum_{i=1}^{n_h} w_i^o \phi(w_i^h \cdot x_j + b_i) - t_j\|_2^2}{N_\nu}}$$

where N_ν is the number of validation parameters.

Also, it has better learning rate than very known methods as BP, SVM and LS-SVM. Three datasets were used for testing (DS-255 with 255 images, DS-160 with 160 images, DS-66 with 66 images) which contain test cases of 256×256 MRI images in axial view. There are 11 different types of diseases which can be put into groups:

- Brain tumor:
 - Glioma, Meningioma, Sarcoma
- Infectious:
 - CTP, MS, HE
- Degenerative:
 - AD, AD plus VA, PD, HD
- Cerebrovascular: CSH

A representation of every case's image can be found on fig.3.10. The classification problem is marked as binary, as the two classes that the samples are separated into are healthy and pathological, whereas pathological is marked anything of the above.

The research team used 5-fold cross-validation (CV) for the statistical analysis of the DS-160 and DS-255 so that overfitting problems will be avoided. A simple interpretation of

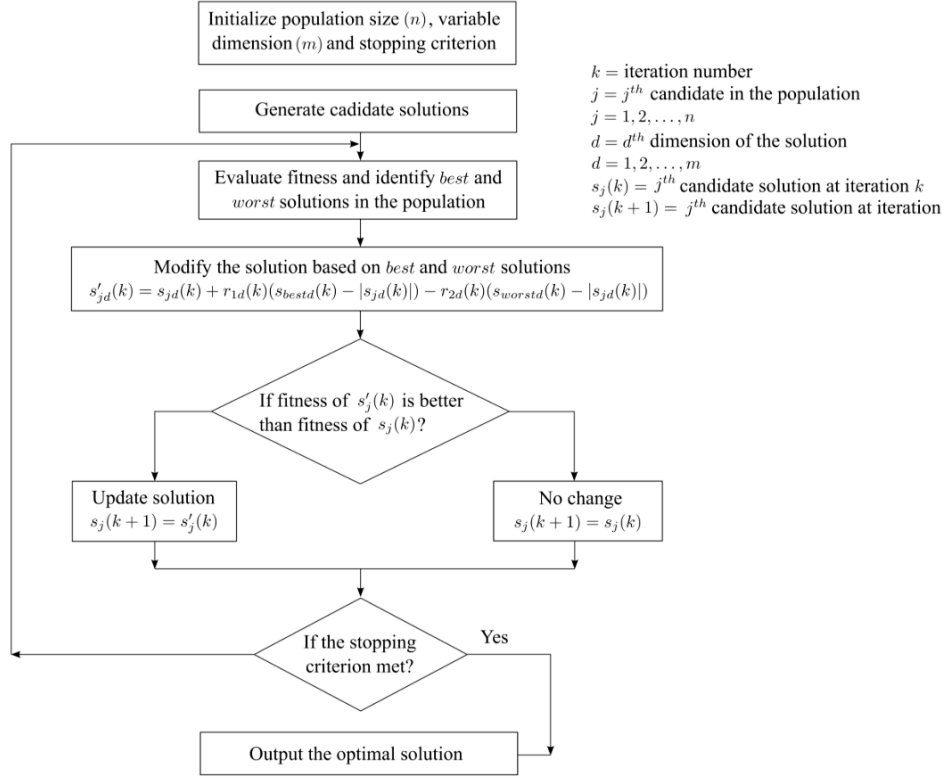


Figure 3.9: Flow diagram of the Jaya algorithm. Figure taken from [6].

that can be seen in fig.3.11. The validation set is there for computation of parameters of the IJaya-ELM method, to pinpoint the moment that further training is insufficient. For DS-66, they employ 6-fold CV, otherwise we would run into instability issues as there are different number of samples from two classes. The final results are an average of 10 different runs of the same datasets in order to remove any randomness on the final observations. On pre-processing, the MR images are divided into 64 regions each, using CLAHE. The clip parameter (β) is set to 0.01 and number of bins is equal to 256. Then, O-DR2T is applied on each input, employing two levels of 2D-DWT, using Haar wavelet as basis. In the end, we have 65536 features, as the images are of size 256×256 . As this is a very big number for the model, we reduce the ones we take advantage of in the end by employing PCA+LDA. We can see how separating PCA from PCA+LDA affects the final Acc in fig.3.13. From this figures, we can also observe that PCA with 15 features and PCA+LDA with two features are the variants with the best results. In fig.3.14 we see a performance comparison between different learning classifiers tested on all three datasets. In IJaya-ELM, they use *sigmoid* function as activation function, normalizing every input in the rang of $[-1, 1]$. The error threshold set for this classifier is 0.05 while acceleration coefficients are set as 2 on both levels, while in E-ELM the

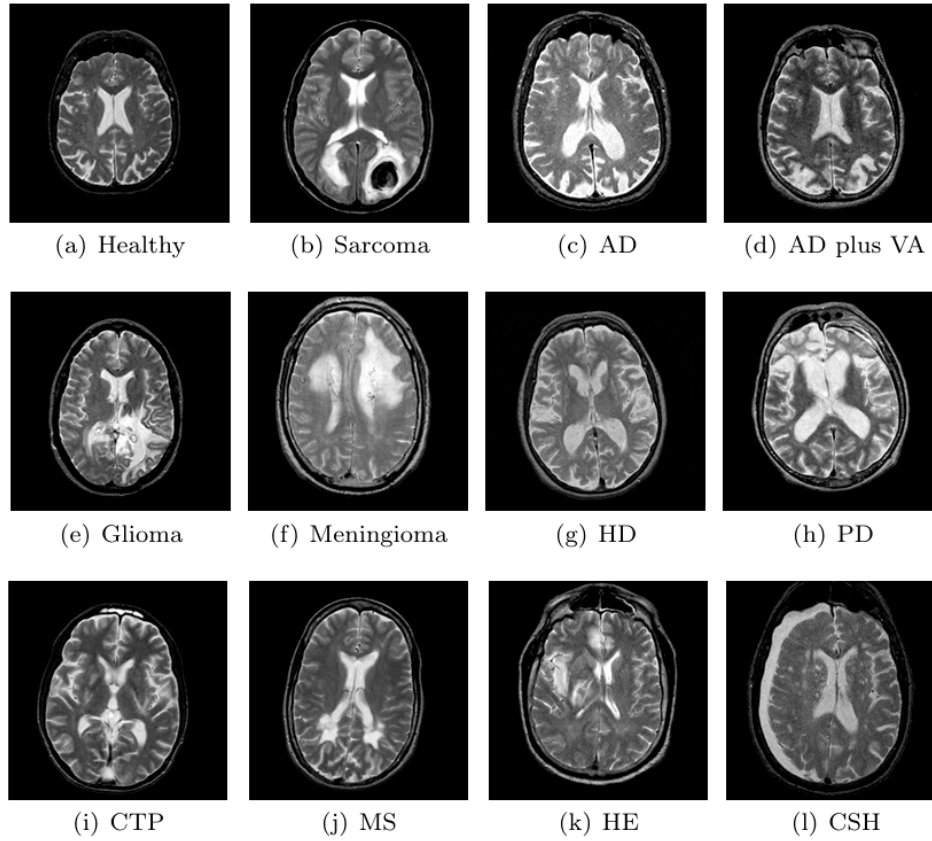


Figure 3.10: Samples of T2-weighted brain MR images.

crossover rate (CR) is set as 0.9 and scaling factor (F) is chosen as 0.8. In APSO-ELM, the initial and final inertia parameters ω_1 and ω_2 are set to 0.4/0.9. For GA-ELM, the CR is set to 0.7 while the mutation rate is set to 0.1 respectively. The IJaya-ELM manages to have the highest Acc of the subset with less hidden neurons. In the other end, Jaya-ELM manages to achieve optimal performance on DS-160 while it falls underneath its less resourceful counterpart on the other two datasets. The norm value reveals that generalization is more efficient on IJaya-ELM compared to anything else while also the smallest norm value is linked to the smallest condition value of matrix H . The results that we see above come after 50 trials.

To test the limits of IJaya-ELM with two features, we compare its Acc to that of k-NN, RF, SVM, BPNN, ELM and Jaya-ELM on all three datasets (fig.3.15). The Acc values are computed for the DS-66 dataset as k-NN (98.94%), BPNN (100%), RF (99.39%), SVM (100%), ELM (100%), Jaya-ELM (99.85%) and IJaya-ELM (100%). On the DS-160, the results are: k-NN (99.44%), BPNN (99.88%), RF (99.56%), SVM (99.88%), ELM (99.94%), Jaya-ELM (100%) and IJaya-ELM (100%). And, last but not least, on DS-255: k-NN (98.88%), BPNN (99.29%), RF (99.02%), SVM (99.33%), ELM (99.37%), Jaya-ELM (99.61%) and IJaya-

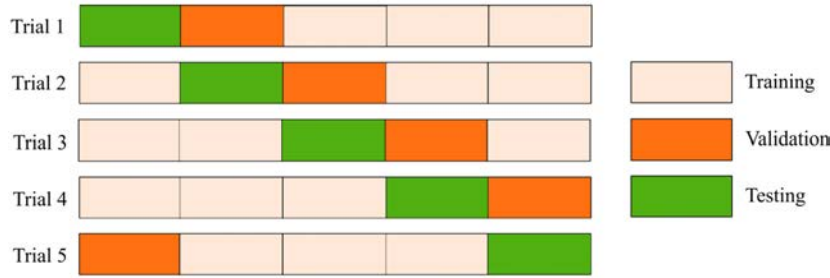


Figure 3.11: Illustration of 5-fold cross validation setting for a single run, as captured in [6].

Dataset	K -fold SCV	Total samples		Training		Validation		Testing	
		H	P	H	P	H	P	H	P
DS-66	6	18	48	12	32	3	8	3	8
DS-160	5	20	140	12	84	4	28	4	28
DS-255	5	35	220	21	132	7	44	7	44

Figure 3.12: Statistical setting of K -fold SCV for three benchmark datasets.

ELM (99.69%). As we understand from the results, the proposed learning algorithm is found the most suitable algorithm along most cases.

The O-DR2T+PCA+LDA+IJaya-ELM on DS-255 dataset manages to classify 2195 out of 2200 samples correctly, misclassifying the rest five as false negative. Out of 350 samples, 347 are classified correctly. The three parameters we mentioned before, sensitivity (S_e), specificity (S_p), Prec values (P_r) of the proposed architecture have values of 99.77%, 99.14% and 99.86% respectively. We can see an analytical comparison on fig.3.16. We see again that on practice the PCA+LDA makes a different as it performs better on every case while also keeping slightly less sensitivity, specificity and Prec. On fig.3.17 we see a bit more clear the impact each improvement brings to the classification.

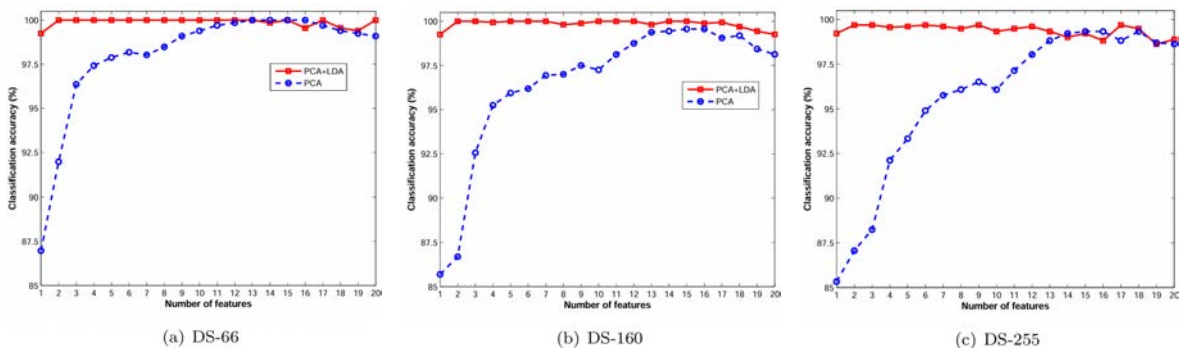


Figure 3.13: Classification accuracy with respect to # of features for the three datasets. Figures are taken from [6].

Classifiers	ACC (%)	Hidden neurons (n_h)	Norm	Condition number (K_2)	Classifiers	ACC (%)	Hidden neurons (n_h)	Norm	Condition number (K_2)
BPNN	100.00	4	-	-	BPNN	99.88	5	-	-
ELM	100.00	5	42.4354	3.5919e+03	ELM	99.94	6	89.1218	5.1528e+03
GA-ELM	100.00	3	32.7753	120.6437	GA-ELM	99.88	3	27.7148	118.4492
E-ELM	100.00	3	22.3902	54.6665	E-ELM	100.00	3	12.9528	56.0894
PSO-ELM	99.70	3	30.3307	82.0004	PSO-ELM	100.00	3	17.2419	81.2390
APSO-ELM	100.00	3	20.7755	49.3035	APSO-ELM	100.00	3	14.2694	57.1167
Jaya-ELM	99.85	3	24.2312	62.5218	Jaya-ELM	100.00	3	14.4950	63.1756
IJaya-ELM	100.00	3	16.7405	31.4217	IJaya-ELM	100.00	3	11.7418	44.3138

Classifiers	ACC (%)	Hidden neurons (n_h)	Norm	Condition number (K_2)
BPNN	99.29	5	-	-
ELM	99.37	6	102.6683	4.0175e+03
GA-ELM	99.53	3	26.0816	137.5234
E-ELM	99.65	3	15.6426	74.2324
PSO-ELM	99.57	3	24.9681	113.3067
APSO-ELM	99.65	3	19.4705	98.0824
Jaya-ELM	99.61	3	18.3876	78.2147
IJaya-ELM	99.69	3	12.8941	54.0704

Figure 3.14: Performance comparison of IJaya-ELM to different classifiers for the three datasets.

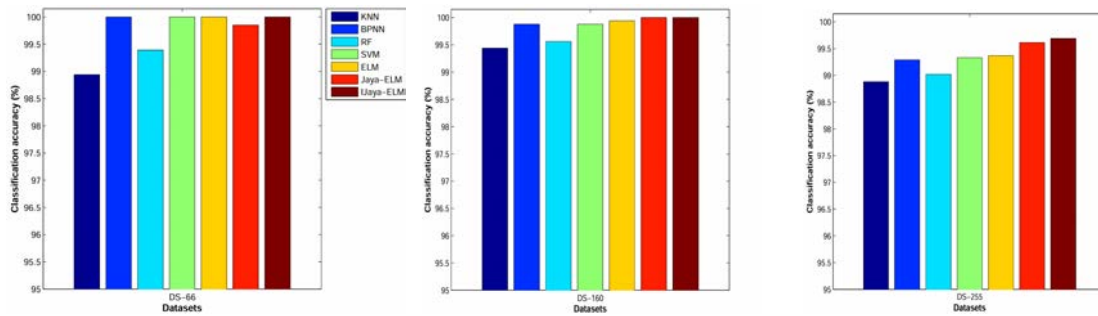


Figure 3.15: Classification accuracy comparison of IJaya-ELM to different classifiers for the three datasets.

3.4 DWA-DNN

In 2019, Mallick et al. [9] published a paper on their bioinformational research, proposing a model that combines DNNs with deep wavelet autoencoder (DWA), combining both the basic feature reduction that autoencoder provides with the segmentation capabilities of wavelet transform.

Autoencoding is based on ANNs and follow an unsupervised process of feature extraction. The wavelet autoencoder has the activation function from wavelet function. Developing a network of WAEs enhances the quality of life of the architecture and provides high level feature learning capabilities. An overview of the suggested model can be seen in fig.3.19. Let's enhance on its parts. The data that are handled to evaluate this architecture are DICOM files, which is a medical file format for computer memory. Because of the available resources

Dataset	Schemes	O-DR2T+PCA+IJaya-ELM	O-DR2T + PCA+LDA + IJaya-ELM
	No. of feature	15	2
DS-66	S_e	100.00	100.00
	S_p	100.00	100.00
	P_r	100.00	100.00
	ACC	100.00	100.00
DS-160	S_e	99.64	100.00
	S_p	99.00	100.00
	P_r	99.86	100.00
	ACC	99.56	100.00
DS-255	S_e	99.50	99.77
	S_p	98.29	99.14
	P_r	99.73	99.86
	ACC	99.33	99.69

Figure 3.16: Classification performances (%) of the proposed schemes based on PCA and PCA+LDA for all three datasets.

Schemes	No. of features	DS-66	DS-160	DS-255
DWT + PCA + IJaya-ELM	15	100.00	99.38	99.17
DWT + PCA+LDA + IJaya-ELM	2	100.00	99.94	99.41
O-DR2T + PCA+LDA + IJaya-ELM	2	100.00	100.00	99.69

Figure 3.17: Classification accuracy (%) of the proposed method with wavelet based method.

in terms of input, the images get split into sub-groups that improves spatial awareness and pushes the performance further. The dataset is the RIDER (Reference Image Database to Evaluate Therapy Response) Neuro-MRI with 19 patients with recurrent glioblastoma who underwent repeat imaging sets on a 2-day period. As described from the source, "All 19 patients had repeat dynamic contrast-enhanced MRI (DCE-MRI) datasets on the same 1.5T imaging magnet. On the basis of T2-weighted images, technologists chose 16 image locations using 5mm thick contiguous slices for the imaging. For T1 mapping, multi-ip 3D FLASH images were obtained using ip angles of 5, 10, 15, 20, 25 and 30 degrees, TR of 4.43 ms, TE of 2.1 ms, 2 signal averages. Dynamic images were obtained during the intravenous injection of 0.1mmol/kg of Magnevist intravenous at 3ccs/second; started 24 seconds after the scan had begun. The dynamic images were acquired using a 3D FLASH technique, using a ip angle of 25 degrees, TR of 3.8ms, TE of 1.8ms using a $1 \times 1 \times 5$ mm voxel size. The 16 slice imaging set was obtained every 4.8sec. Seventeen of the 19 patients also obtained repeat diffusion tensor imaging (DTI) sets. Whole brain DTI were obtained using TR 6000ms, TE 100ms, 90 degree ip angle, 4 signal averages, matrix 128×128 , $72 \times 172 \times 5$ mm voxel size, 12 tensor directions, iPAT 2, b value of 1000 sec/mm^2 . All 19 patients underwent whole brain 3D FLASH imaging in the sagittal plane after the administration of Magnevist. For this sequence, TR was 8.6ms, TE 4.1ms, 20 degree ip angle, 1 signal average, matrix 256×256 ; 1mm isotropic voxel size. All 17 patients who had repeat DTI sets also had 3D FLAIR sequences in the sagittal plane

Existing PBDs	Feature size	Run	ACC (%)		
			DS-66	DS-160	DS-255
DWT + SVM + POLY [5]	4761	5	98.00	97.15	96.37
DWT + PCA + BPNN + SCG [53]	19	5	100.00	98.29	97.14
DWT + PCA + FNN + SCABC [61]	19	5	100.00	98.93	97.81
DWT + PCA + FNN + ACPSO [59]	19	5	100.00	98.75	97.38
DWT + PCA + KSVM [60]	19	5	100.00	99.38	98.82
DWPT + SE + GEPSVM [52]	16	10	99.85	99.62	98.78
DWPT + TE + GEPSVM [52]	16	10	100.00	100.00	99.33
WPTE + FNN + RCBBO [38]	16	10	100.00	100.00	99.49
WE + HMI + GEPSVM [58]	14	10	100.00	99.56	98.63
DWT + PCA + ADBRF [27]	13	10	100.00	99.18	98.35
FRFE + WTT + TSVM [43]	12	10	100.00	99.69	98.98
DTCWT + VE + GEPSVM [39]	12	10	100.00	99.75	99.25
FRFE + WTT + DP-MLP + ARCBBO [55]	12	10	100.00	99.19	98.24
RT + PCA + LS-SVM [8]	9	5	100.00	100.00	99.39
DWT + PCA + k -NN [11]	7	5	98.00	97.54	96.79
FPCNN + DWT + PCA + FNN [10]	7	10	100.00	98.88	98.43
SWT + PCA + IABAP-FNN [42]	7	10	100.00	99.44	99.18
SWT + PCA + ABC-SPSO-FNN [42]	7	10	100.00	99.75	99.02
SWT + PCA + GEPSVM [51]	7	10	100.00	99.62	99.02
WE + NBC [67]	7	10	92.58	91.87	90.51
DWT + PCA + LDA + RF [25]	7	10	100.00	99.75	99.14
MBD + SLFN + PSO-TTC [63]	5	10	100.00	98.19	98.08
O-DR2T + PCA + IJaya-ELM	15	10	100.00	99.56	99.33
O-DR2T + PCA+LDA + IJaya-ELM (Proposed)	2	10	100.00	100.00	99.69

Figure 3.18: Comparison analysis with other competent PBDs on three standard datasets.

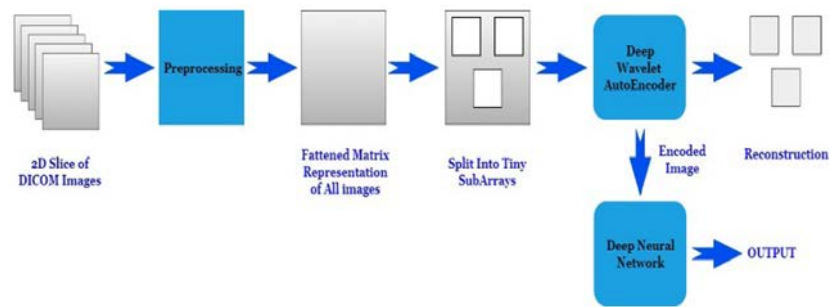


Figure 3.19: Proposed architecture of a DICOM image classifier for brain disease detection based on DWA-DNN model, as designed in [9].

after the administration of Magnevist. For this sequence, the TR was 6000ms, TE 353ms, and TI 2200ms; 180 degree ip angle, 1 signal average, matrix 256×216 ; 1 mm isotropic voxel size. Before transmission to NCIA, all image sets with 1mm isotropic voxel size were defaced using MIPAV software or manually.” All the images collected are in DICOM format and they have been processed using python program and the total size of the image dataset is 7.3GB. The sample image dataset is shown in figure 3.20. After follows the DWA that results into encoded images and, finally, the images that get used are those that have gone through approximation for the processing of the DNN.

Optimizing DNNs can be done by a number of methods, one of them being autoencoders. It achieves feature extraction from large data. Here, it is a deeper network that handles image

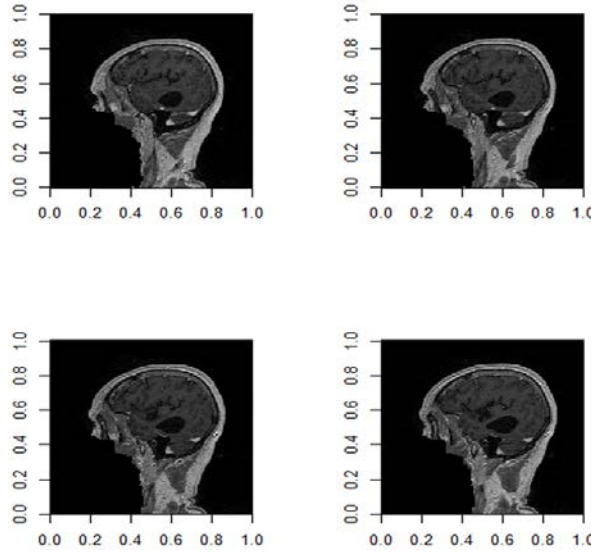


Figure 3.20: Sample image of the RIDER dataset, used in the DWA-DNN method[9].

compression to pre-process the labels of the data. An interpretation of an autoencoder can be seen in fig.3.21. A closer interpretation of the autoencoder used in this case can be seen in

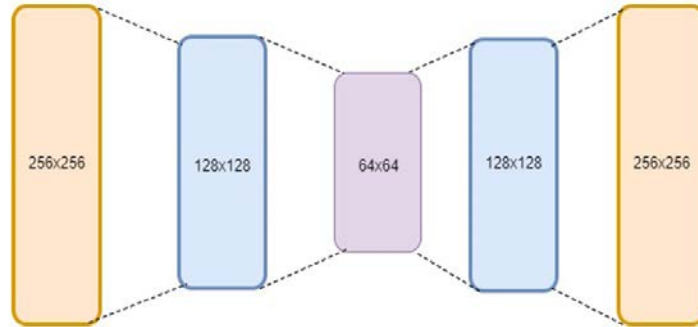


Figure 3.21: Simple autoencoder model with 3 hidden layers for encoding and decoding of images.

fig.3.22. The middle layer that contains the encoded data is of size 64×64 . The activation functions is:

$$h_i = f_i(W_i X_i + b_i), i = [1, 4]$$

where X_i is the input, H_i depicts the hidden layer ($[1, 3]$) and Y_i is the output. W_i is the weights vector between X_i , the H_1 to H_2 and Y_1 .

There are different types of autoencoders that apply to different algorithms. One of them is sparse autoencoder that works with supervised, scalable learning as it has a high number of hidden neurons, higher than that of the input neurons which means that only a number of the hidden units are active at the same time. Sparse autoencoders don't have prepared data on

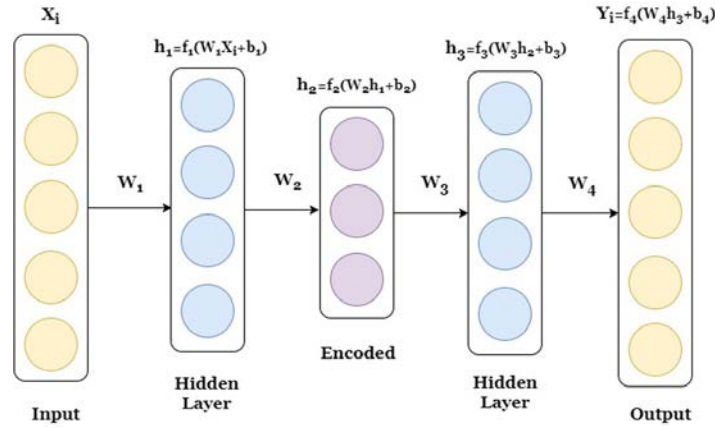


Figure 3.22: Autoencoder model with different layers, functions and parameters.

training. It works by constructing a dictionary from the input that improves the classification by scaling down the error rate on the reconstruction of the language. Rather than sparsifying the data, it stores everything in a single class that improves the prediction. To put it behind maths, the formulation is depicted as:

$$X = f(W_v + s)$$

$$v' = f(W^t X + b')$$

An encoding step is when a single hidden layer (H) is connected to the input vector (v) with a weight matrix (w). The final prediction on the output (v') is reconstructed with the help of a new weight matrix (w^t). s represents the bias vector while f is the activation function. The learning process for the error propagation is stated as:

$$\min ||v - v'||_2^2$$

Next autoencoder that we discuss is the de-noising variant. Since DNNs have a nonlinear nature, artificial noise is important on pre-training on each layer. This gives the hidden layer the ability to reconstruct corrupted data and lessen the error when rebuilding the dictionary. This process can be seen in figure 3.23. The training of the de-noising autoencoder is simple and it is done by manipulating the dataset, decreasing its validity and feeding it to the NN. Finally, the last autoencoder that we discuss is the deep wavelet autoencoder or DWA. As we mentioned before, this autoencoder makes the model deeper. The image goes through Discrete Wavelet Transform (DWT) using Daubechies mother wavelet of order 2 to get approximate and detail coefficients by passing through low pass and high pass filters respectively. From these, only the approximation coefficients are used further for classification.

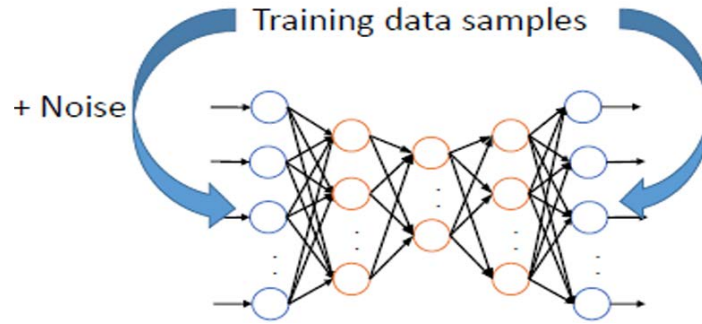


Figure 3.23: Overview of de-noising autoencoder.

The experiment is done on Python 3.6 with the mandatory packages (e.g. numpy, scipy, matplotlib, keras, scikit-learn, tensorflow) on an i7 core processor, 3.4GHz speed with 4GB RAM. The activation function chosen for the DNN is the *Sigmoid* function with learning rate of 0.8 with 1000 epochs in total.

Let's present the overview of the algorithm 2.

Algorithm 2 Deep wavelet autoencoder-based DNN

- 1: Pre-processing of DICOM to extract the specific image matrix only.
 - 2: Flattening of image matrices to construct image dataset.
 - 3: Splitting of dataset to sub-arrays
 - 4: **for** Each sub-array **do**
 - 5: Input the image sub-array to Deep Wavelet Autoencoder for encoding.
 - 6: Pass the encoded image through low pass and high pass filter using discrete wavelet transform for decomposition.
 - 7: Apply inverse wavelet transform to combine and decode the images to get original image.
 - 8: Run Autoencoder for # of epochs to get optimized weight and bias.
 - 9: Extract approximatipn coefficients from the hidden layer, combine them and provide as input to a DNN for classification.
 - 10: Train the DNN with the inputs provided by 9. and test the network for different metrics measurement.
 - 11: **end for**
-

Taking a look on the results show on the table of fig.3.24(α'), we find four different metrics for performance: a) Acc, b) specificity, c) sensitivity and d) F-score. Also, on fig.3.24(β'), we see an experimental comparison between the DWA-DNN, the DNN on its own and the AE-DNN itself. The proposed implementation outperforms the typical classifiers found in most models. All metrics show the advanced performance of the newly proposed architecture on

a 10-fold cross validation.

The impact of the Wavelet transform can be seen on the comparison graph found on fig.3.25.

Overall, the model showed 93% Acc, 93% F-score, 94% sensitivity and 92% specificity.

Classification Techniques	Accuracy	Specificity	Sensitivity	F-Score
MLPNN	0.85±0.33	0.83±0.26	0.87±0.22	0.84±0.30
RBFNN	0.67±0.22	0.75±0.23	0.74±0.34	0.74±0.21
ELM	0.90±0.15	0.87±0.32	0.91±0.22	0.89±0.25
PNN	0.89±0.18	0.90±0.28	0.87±0.29	0.88±0.32
TDNN	0.86±0.32	0.85±0.25	0.88±0.23	0.86±0.29
DWA-DNN	0.93±0.14	0.92±0.16	0.94±0.26	0.93±0.15

Classification Techniques	Accuracy	Specificity	Sensitivity	F-Score
DNN	0.89±0.18	0.88±0.26	0.91±0.19	0.90±0.22
AE-DBN	0.90±0.19	0.89±0.24	0.91±0.18	0.90±0.23
DWA-DNN	0.93±0.14	0.92±0.16	0.94±0.26	0.93±0.15

(α') Performance comparison between DL vs. non DL approaches.

(β') Performance comparison between DNN, AE-DNN and proposed DWA-DNN.

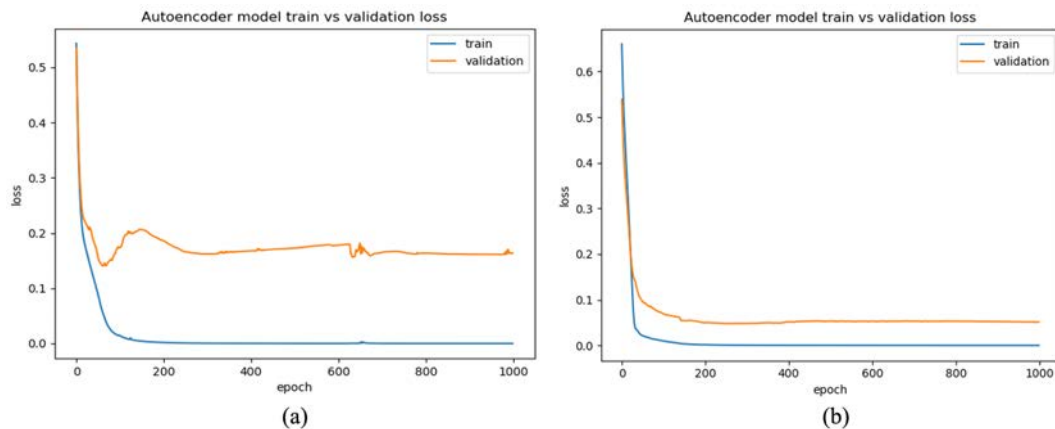


Figure 3.25: Loss graph for Autoencoder model. (a) Simple AE model. (b) Wavelet AE model. Graphs taken from [9].

3.5 CNN-SVM TH

In 2022, Kharandish et al.[10] published their AI implementation on tumor detection and classification that is based on NNs. The proposed hybrid models is built combines CNN and SVMs and its based on MR image segmentation to result to classification. If we take a look on figure 2.3, we see a CNN, layer by layer, built to work on image analysis. In general, CNNs start with the input layer, then continue to the hidden layer(s) and lead to output layer.

The process that the input goes through is that, first, the convolutional layer extracts some features, as each pixel produces information both about itself and the neighbor pixels. Their

values present the mathematics behind their correlation. Next is pooling, which clears up the data needed from those that can be avoided and reduces the complexity of the data. Pooling features can introduce overfitting issues, in which case, the flatten layer gets employed to convert 2D arrays to 1D array before we reach the fully connected neurons. The other half of the model consists of an SVM, a supervised ML that replaces the last layer to improve the effectiveness of the kernel. In more detail, what comes with this change is that we have access to add hidden layers which makes the model more accurate and faster on the features extracted. This connection between the two is clear on fig.3.26, where we see the total structure of the model. At first, there is a basis of 28×28 feature map with 5×5 convolution that leads to a 14×14 feature map, accommodated with 2×2 convolution matrix. After feature processing is done, SVM takes the labels found and trains the feature vectors, classification and decision making in the better way. In reality, the improvement can be a critical factor on reducing the mortality rate caused by brain tumors.

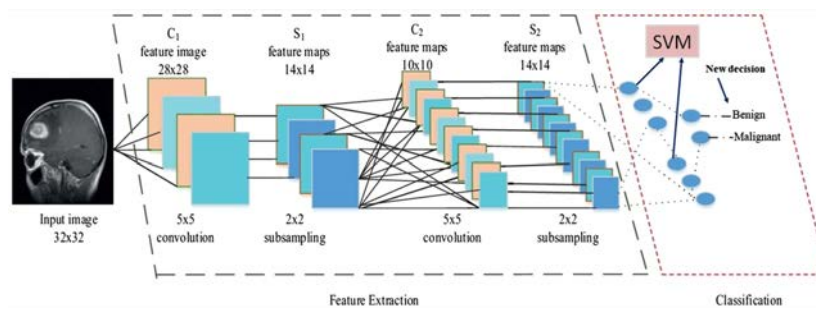


Figure 3.26: The structure of CNN-SVM TH, as interpreted in [10].

There are five essential stages of the proposed methodology. These are:

- **MRI introduction:** Input the image of a brain that will be processed.
- **Pre-processing:** There are three subsections of the image pre-processing:
 - Resizing: The image has to be resized to a specific format, so that all data can be processed in equal matter.
 - Skull removal: The skull is considered as the background of the important information, so it must be removed.
 - Filtering: The median filter gets applied to remove noise for better classification Acc.

- **Feature extraction:** From a plethora of different algorithms, the researchers decided to recommend a number of them, starting with the Henry gases solubility optimization (HGSO) algorithm[11] for feature extraction. What lead this decision was the ability to evaluate different block sizes of image features to find the best factor to find a similar region to the query image. Another one was proposed by Essam et al. with the Harris hawks optimization (HHO)[12] which uses KNN and SVM for the same reasons, showing great expectations. A different approach was taken by Abdel-Azim et al.[13], as they proposed binary optimization techniques that depend on the hyperbolic angle of fitness equation of an image to manage the best feature choice.
- **Image segmentation:** Partitioning the image into different sections helps with pattern recognition. A threshold value and max value are needed to enhance the severity of this process. The threshold decides if the grayscaled image will have white or black blocks.
- **Classification:** Starting with training both CNN and SVM, we then move to evaluating the SVM individually, CNN individually and the hybrid SVM/CNN combination with *Positive predictive value (PPV)*, *Feature point vector (FPV)* and *Acc* evaluation.

Before moving to the evaluation, we should discuss the metrics that we'll take into consideration to compare the proposed model to the other implementations. These are VAR, the mean brightness level of the image, EN, ASM, HOM and contrast.

The values of these statistical functions for the current case can be seen in figure 3.27. The dataset is also a familiar one from before, as the information has been taken from BRATS

Comparison of statistical functions based on Benign and Malignant.					
Sl. No.	Feature	Range for Malignant (Min)	Range for Malignant (Max)	Range for Benign (Min)	Range for Benign (Max)
1	Mean	73.1	100.2	45.06	54.58
2	Variance	74.09	114.46	78.52	93.22
3	Standard Deviation	8.6	10.69	8.86	9.65
4	Energy	0.19	0.14	0.21	0.38
5	Entropy	7.3	7.5	5.87	6.66
6	Homogeneity	0.94	0.88	0.47	0.9
7	Contrast	0.11	0.46	0.44	0.47
8	Correlation	0.98	0.94	0.89	0.93

Figure 3.27: Comparison of statistical functions based on Benign and Malignant. The values were taken from [10].

2015. The setup used here was an Intel Core i7 CPU, 8GB of RAM and 4GB Nvidia GPU and the evaluation is done on **Acc**, **PPV** and **FPV**.

The outcome of the hybrid model was very promising, as the overall Acc of the CNN-SVM model was 98.495%, with benign classification of the SVM counterpart being 61.67% accurate but by adding the CNN which got a 97.724% result, we reach the final 98.5873%. The malignant tumor classification was 67.98% accurate on SVM, CNN got 97.8455% and the hybrid model reached 98.6702%. These results can also be seen summarized in fig.3.28. The overall Acc taken from this study is 72.5536% on SVM, 97.4394% on CNN and 98.4959%. Taking a closer look, even though the separate Acc of the two sides seems to not have significant progress, the combination of the two expresses progress as cases pass.

Looking on the FPV figures, we find similarities to the Acc, as on their separate analyses, both sides don't show improvements, while the hybrid implementation produces lower values.

PPV shows the rate of correct classification of the image. Both CNN and SVM don't show satisfactory results when processed on their own. The CNN-SVM architecture takes a leap, showing great improvement in the correction of the classification of the dataset.

In figure 3.30 we see a comparison of the hybrid model to other existing proposed systems. The newly employed hybrid evaluation method shows great results, advancing the standard in the industry and creating hopes for successful diagnosis of brain tumors.

Proposed hybrid model classification accuracy			
Stage	SVM	CNN	Hybrid
Benign	61.67%	97.724%	98.5873%
Malignant	67.98%	97.8455%	98.6702%

Figure 3.28: Comparison of accuracy between the different steps of the CNN-SVM TH hybrid model.

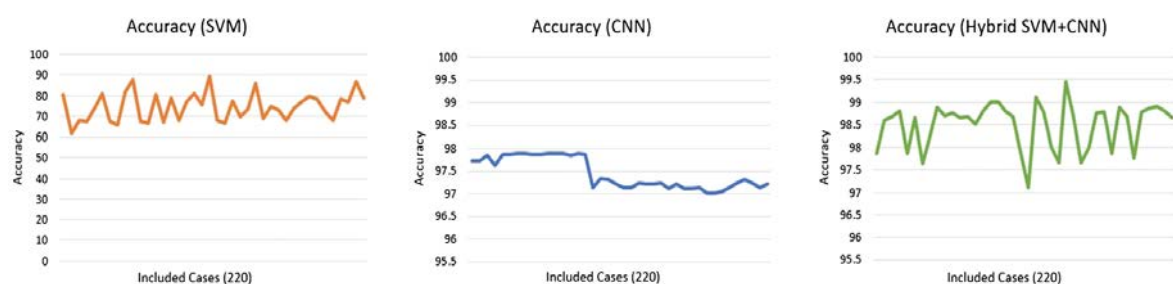


Figure 3.29: Accuracy comparison of CNN, SVM with Hybrid CNN-SVM TH model.

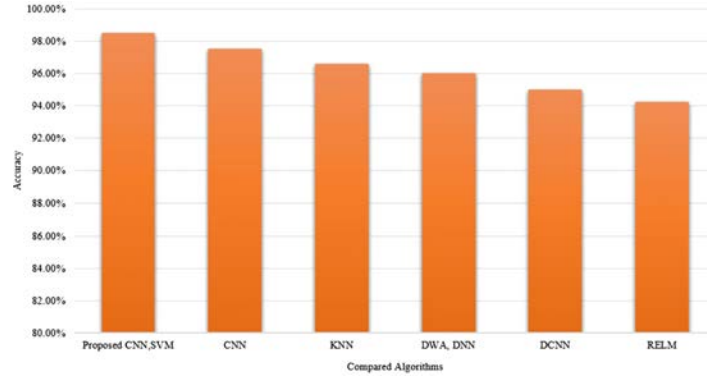


Figure 3.30: Comparison of the hybrid CNN-SVM TH model to existing methods.

3.6 CNN-NADE

Rahekeh et al.[14] proposed a method to detect three different types of brain tumors (pictuary, glioma and meningioma) (3.1) that is based on the combination of CNN with a NADE (Neural Autoregressive Distribution Estimation) model. The classification is done again on MRI images for tumor detection.

CNNs is something that we have mentioned excessively before, as it is widely used on classification models as the one we are discussing now. On the other hand, NADE is something that we haven't considered seen yet. Let's discuss how it works.

Considering input $x \in R^D$, where D is the dimension of samples. It has autoregressive services, as the autoencoder weights are set by masks which remove the possibilities of vain representations and enable accurate koint distribution for the network. The valid loss function can be described as:

$$l(x) = -\log(p(x_1, \dots, x_D)) = \sum_{d=1}^D -\log(p(x_d | x_{<d}))$$

$$= \sum_{d=1}^D -x_d \log(p(x_d = 1 | x_{<d})) - (1 - x_d) \log(p(x_d = 0 | x_{<d}))$$

Another service accomplished by the model is handling real valued data, as it introduces Gaussian elements to the information:

$$p(x_{0_d}, x_{0_{<d}}) = \sum_{c=1}^C \pi_{0_d c} N(x_{0_d}; \mu_{0_d, \mu}, \sigma_{0_d, c}^2)$$

where:

$$\pi_{0_d, c} = \frac{\exp\{z_{0_d, c}^{(\pi)}\}}{\sum_{c=1}^C \exp\{z_{0_d, c}^{(\pi)}\}}$$

$$\begin{aligned}
\mu_{0,d,c} &= z_{0,d,c}^{(\pi)} \\
\sigma_{0,d,c} &= \exp\{z_{0,d,c}^{(\sigma)}\} \\
z_{0,d,c}^{(\pi)} &= b_{0,d,c}^{(\pi)} + \sum_{k=1}^H V_{0,d,k,c}^{(\pi)} h_{d,k} \\
z_{0,d,c}^{(\mu)} &= b_{0,d,c}^{(\mu)} + \sum_{k=1}^H V_{0,d,k,c}^{(\mu)} h_{d,k} \\
z_{0,d,c}^{(\sigma)} &= b_{0,d,c}^{(\sigma)} + \sum_{k=1}^H V_{0,d,k,c}^{(\sigma)} h_{d,k}
\end{aligned}$$

Building a convolutional NADE gives the system an autoregressive model that handles feature extraction from raw high dimensional data because of the convolution layers and has autoregressive characteristics due to the NADE part of it.

The dataset that the researchers decided to use uses contrast-enhanced MRI imaging with 3064 T1-weighted samples from 233 patients. 708 are diagnosed with meningioma, 1426 with glioma and 930 with pituitary brain tumors. Each image is a square of 512 pixels on each dimension that gets reduced to a 64×64 image to decrease the computations needed by getting rid of unneeded pixels. The method was written on Python 3.6.9 with the help of Tensorflow and Keras libraries. The testbench includes a CPU and discrete GPU (e.g. Nvidia K80s, T4s, P4s, P100s), 12.8GB of RAM on Colab and 33GB of hard disk.

We can see the network's architecture on fig.3.31, which consists of two CNNs, a distribution estimator and a fully connected network. Training the NADE model improves the autoregressive prediction Acc and the de-noising of the data. The researchers also implement two CNNs that are learning separately to make sure that we prevent the noise introduced by the raw data. The activation function for the hidden layer and output layer are *ReLU* and *Softmax* respectively while training has a cross-entropy loss function:

$$H(y, p) = - \sum_i y_i \log(p_i)$$

The CNN contains three convolutional layers of eight, sixteen and eight channels with filter size of five, three and three with pooling size of two. They, also, employ 6-fold cross-validation due to the lack of samples.

The Acc that the authors reached on a single CNN was 92.74%, which improved to 93.14% when the second one was added. Moving to the last form of the architecture, the Acc of 95% is obtained. As a measurement, we also take the F1-score of the final result.

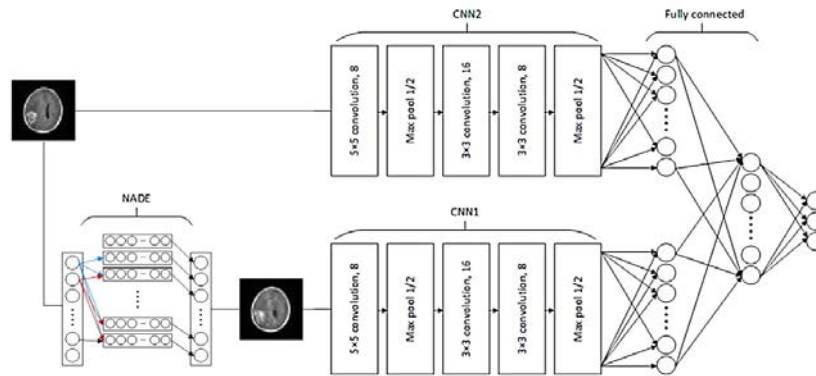


Figure 3.31: Network's architecture of CNN-NADE method [14]. This structure contains two CNNs and a density estimator.

This model was measured with 0.9464 Sens, 0.9742 Spec and 0.9449 Prec which results to 0.9456 F1-score, with a more detailed look in fig.3.32. An interesting observation noted on the original paper is that even though the MR images of meningioma tumor are the ones with the least cases, it reaches at least 1% higher Acc compared to other models. When comparing the current model to other proposed in recent years, we understand that even though in terms of performance it's on the back foot by a small percentage (around 1%), the fact that it achieves such results with a low computational overhead makes it a great solution to implement on a larger scale. Of course this gap to higher Acc can prevent a false prediction, saving a life, but we can understand that with a few further optimizations, improving the outcome is a matter of time. We can understand that the low computational logic also means that there's more space left for improvements.

	CNN	Cascade model	CNN-NADE
meningioma	0.887	0.884	0.898
glioma	0.947	0.945	0.952
pituitary	0.980	0.986	0.985

Figure 3.32: F1-score for the three stages of the proposed CNN-NADE method [14].

3.7 CNN T1-CE

Moving forward, we find a very similar method to the previous one, published in 2020 from the University of Belgrade, lead by Badza and Barjaktarovic [15]. It handles the classification of the same three types of tumor (as seen in fig.3.1). What makes a difference in this case is the training of the model, as it is done in T1-weighted contrast-enhanced magnetic

resonance images. The detailed information that the dataset can provide on training means that the model itself can be even more simple, decreasing the scale of computational logic required to build it. Let's discuss this implementation, step-by-step.

As we have said in previous sections, MRI is the main technique of diagnosing a tumor and the type of it. However, the samples on their own don't give enough information, as it isn't easy for a human to observe every small feature found in this low-resolution sample, thus the need of technology in this field. In fact, biopsy of brain tumor has to be done through surgery, risking both the patient's health and the clinic's wealth. This flags the urgency of an effective way of diagnosis of such matter.

Starting from the database, the authors chose to go with dataset published in 2017 by Nanfang Hospital and General Hospital, Tianjin Medical University, China that includes 3064 T1-weighted CE-MRI samples that were collected from 2005 to 2010 [16]. There three different typed of tumors found: 708 images of meningioma, 1426 images of glioma and 930 images of pituitary tumor coming from 233 patients, on three different planes: sagittal (1025 images), axial (994 images) and coronal (1045 images).

Moving to pre-processing of the data, they first normalized and resized the image to 256×256 pixels from the initial *int16* format. While also cloning the dataset, rotating it by 90° , flipping images vertically, which means that we multiply the data the we can process to three times the initial ones, resulting to 9192 samples.

The architecture itself was developed in MATLAB R2018a and consists of the input vector, two main blocks, where the first one consists of a RELU based convolution layer that halves the initial input, feeding it to a pooling layer, which then goes to the second convolution block that doesn't diminish the dimensions further and leads to the classification block and last is the output layer. In total, the hidden layers are two of the first block, two of the second, the classification block and the output layer. A visual interpretation is found in figure 3.33. The

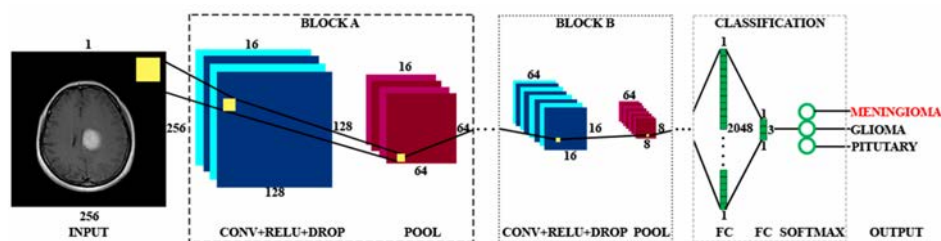


Figure 3.33: Schematic representation of the CNN T1-CE architecture proposed by Badza and Barjaktarovic [15].

performance of the network is measured by two k-fold cross-validation method, first being a 10-fold record-wise cross-validation, meaning that we randomly divide the data into 10 approximately equal portions so that each tumor category was equally present in each portion. The second is subject-wise variant, which means that now separates the input into 10 sectors, each one containing a single subject. In order to make the implementation more accurate, we separate the training data to testing data in order to generalize the training process of the model and a realistic image of the results we would get in the industry. Six portions of the dataset are used for training and two portions for testing. Training was done through Adam optimizer, choosing a mini-batch size equal to 16 with data shuffling enabled. Training finishes only when the loss value starts to rise. Regularization is set to 0.004 and initial learning factor rate equal to 0.0004. Training and testing was done on an Nvidia GeForce GTX 1050 Ti GPU.

The results of this implementation can be seen in figure 3.34, while also we see the supplementary visualisations of the results on their respective figures (3.35(α') - 3.35(δ')). To understand better confusion matrices, non-white rows represent network output classes while non-white columns correspond to real classes. Correctly classified images are shown on the diagonal. Bottom right corner shows the overall Acc. On the table, we see the test Acc, Prec, Rec and F1-score per test case.

For the record-wise 10-fold cross-validation approach, on the original testing case the error recorded is equal to 4.6%, while on the augmented dataset it further improves by 1.2% (figures 3.35(α'), 3.35(β')).

On the subject-wise 10-fold approach, the original dataset had a classification error rate equal to 15.7% which shows that the record-wise variant made way better prediction, expected as the data tested are unknown to the model. In same fashion, the augmented dataset approach performed better but also not as well as the record-wise classification, reaching 88.5% Acc.

Division/Dataset	Testing Approach	Test Accuracy [%]	Average Precision [%]	Average Recall [%]	Average F1-Score [%]
record-wise/the original dataset	10-fold	95.40	94.81	95.07	94.93
	One test	97.39	95.44	96.94	96.11
record-wise/the augmented dataset	10-fold	96.56	95.79	96.51	96.11
	One test	97.28	97.15	97.82	97.47
subject-wise/the original dataset	10-fold	84.45	81.40	82.72	81.86
	One test	90.39	85.99	85.84	85.91
subject-wise/the augmented dataset	10-fold	88.48	86.48	87.82	86.97
	One test	91.84	83.94	82.18	81.78

Figure 3.34: Results from testing the network with 10-fold cross-validation and one test, with two different cross-validation methods and different datasets. Values were taken from [15].

Confusion matrix - testing data

Outputs	meningioma	1270 92.0%	101 7.3%	10 0.7%	92.0% 8.0%
	glioma	108 3.8%	2744 95.5%	20 0.7%	95.5% 4.5%
	pituitary	36 1.9%	7 0.4%	1830 97.7%	97.7% 2.3%
		89.8% 10.2%	96.2% 3.8%	98.4% 1.6%	95.4% 4.6%
		meningioma	glioma	pituitary	
		Targets			

(α') Confusion matrices for the original dataset with record-wise 10-fold cross-validation for testing data.

Confusion matrix - testing data

Outputs	meningioma	3827 95.9%	136 3.4%	27 0.7%	95.9% 4.1%
	glioma	291 3.3%	8384 96.5%	17 0.2%	96.5% 3.5%
	pituitary	127 2.2%	35 0.6%	5536 97.2%	97.2% 2.8%
		90.2% 9.8%	98.0% 2.0%	99.2% 0.8%	96.6% 3.4%
		meningioma	glioma	pituitary	
		Targets			

(β') Confusion matrices for the augmented dataset with record-wise 10-fold cross-validation for testing data.

Confusion matrix - testing data

Outputs	meningioma	873 74.0%	249 21.1%	57 4.8%	74.0% 26.0%
	glioma	334 10.9%	2620 85.7%	103 3.4%	85.7% 14.3%
	pituitary	186 9.8%	35 1.8%	1684 88.4%	88.4% 11.6%
		62.7% 37.3%	90.2% 9.8%	91.3% 8.7%	84.3% 15.7%
		meningioma	glioma	pituitary	
		Targets			

(γ') Confusion matrices for the original dataset with subject-wise 10-fold cross-validation for testing data.

Confusion matrix - testing data

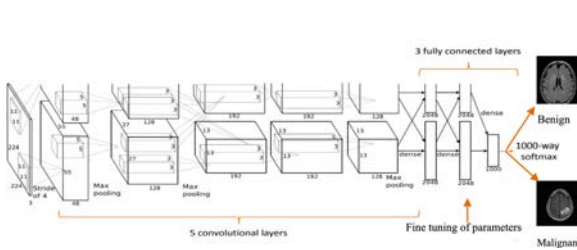
Outputs	meningioma	3077 83.3%	496 13.4%	119 3.2%	83.3% 16.7%
	glioma	848 4.7%	7901 88.8%	149 1.7%	88.8% 11.2%
	pituitary	374 6.7%	114 2.0%	5129 91.3%	91.3% 8.7%
		71.6% 28.4%	92.8% 7.2%	95.0% 5.0%	88.5% 11.5%
		meningioma	glioma	pituitary	
		Targets			

(δ') Confusion matrices for the augmented dataset with subject-wise 10-fold cross-validation for testing data.

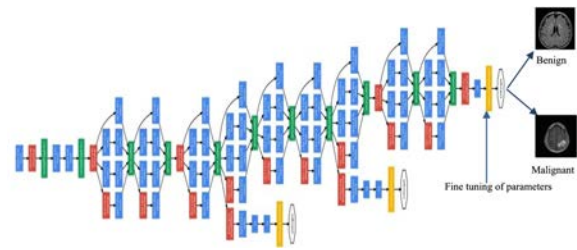
The results obtained from the best classification of the proposed method outperform the rest of the implementations mentioned. We see that the results are comparable, as the difference between the best and worst variant is at 2.79%, not a huge different, considering that the implementation of Gumaie et al.[17] uses 5-fold cross-validation, meaning a more cost-efficient architecture compared to the study we are currently discussing. The second table shows the importance of the k-fold cross-validation, as this extra step that they make the model go through improves the performance severely, as from an Acc around 93%, the best case reached is 97.39% with the original dataset. This means the segmenting the dataset correctly makes a difference on the final outcome. Lastly, the augmented dataset testing gives as Prec of approximately 97%, while Sultan et al.[18] implementation has 1% less.

3.8 DNN-TL

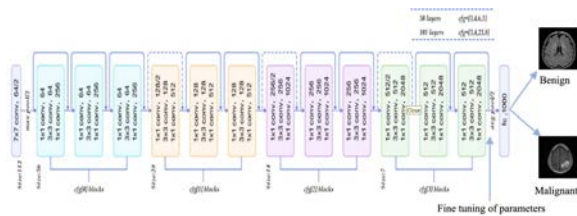
Mehotra et al.[19] proposed their own take on a quick diagnosis method for brain tumors on 2020, adding to the already aforementioned AI techniques proposed by various sides. Their vision is to enhance the exactness of brain MRI classification by using DL algorithms by **Transfer Learning** (TL). What transfer learning achieves is to take the data provided by pretrained systems and utilize them for the new data that can be provided in the future. The DL models that the researchers experimented with are AlexNet (3.36(α')), GoogleNet (3.36(β')), ResNet50, RestNet101 (3.36(γ')) and SqueezeNet (3.36(δ')). Classification is done through the architecture shown in fig.3.37 where we see that we take the input of from the dataset, loading the labels of it and manipulating the provided information (we'll discuss it further later), then moving to separating these data in three segments (70% for training, 15% validation and testing) and then moving to the DL-based extraction of features and classification.



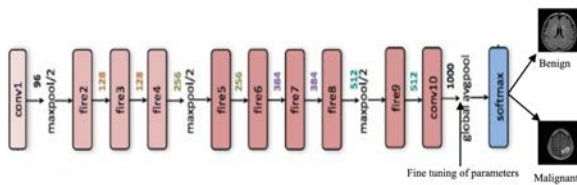
(α') AlexNet architecture.



(β') GoogleNet architecture.



(γ') ResNet architecture.



(δ') SqueezeNet architecture.

The dataset preferred by the research team was The Cancer Imaging Archive (TCIA) public access repository from 2013. From 20 patients, it contains 696 JPEG MR images of 225×225 size, 32% of them labeled as benign while the rest are malignant. Pre-processing starts from downscaling the initial images by half (from squared 512 to squared 255) to lessen the classification delay. Then follows the segmentation of the data into the three parts we mentioned before and, lastly, editing these samples with methods as flipping, mirroring, rotating and adding noise in order to create pseudo-samples for overfitting purposes.

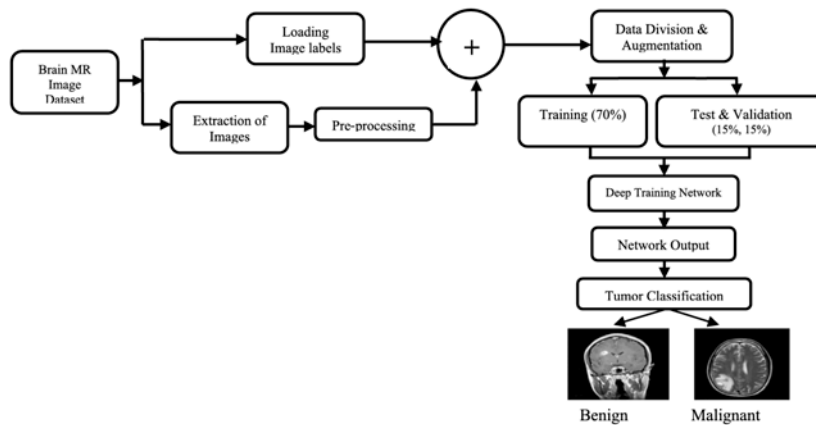


Figure 3.37: The proposed structure of the DNN-TL approach [19].

As we also mentioned, extraction of the features is done through DL networks to make the process faster. The convolution layers are handled by deciding on a precise value of stride so that the progression from one epoch to the next one is done correctly. At times, zero paddings are required if the filter in the convolution later does not cover all images given in the input. They also use ReLU in the activation layer to restrain every negative value to zero in the feature map. Next follows the pooling layers to create non-overlapping area and achieve downsizing. The normalization layer is next, regulating the network and accelerating training. Last is the fully-connected layer which classifies by using a *Softmax* layer with cross-entropy.

The model follows the norm in terms of architecture up until now. What makes it different from the rest, though, is the transfer learning. This approach involves leveraging the knowledge gained from one task to address similar tasks. It entails using the insights from a pre-trained model to develop new models with new data. Fine-tuning a pretrained model with TL is generally faster and easier than building one from scratch. Using pretrained DL systems activates quickly adapting to new tasks. Many researchers and data scientists view TL as a powerful tool that can accelerate progress in AI. There is a clear difference between the traditional approach to training ML models and using a technique based on Transfer Learning (TL) principles. Conventional learning is isolated and specific to particular tasks or datasets, training standalone models without retaining information for future use. In contrast, TL allows you to use knowledge (such as weights and features) from previously trained models, which can be particularly useful for addressing new tasks with limited data. A scheme that describes this approach can be seen in fig.2.1. DL frameworks as the ones we mentioned

before have the ability to learn various features from various levels that are connected to the fully connected layer in the end. Thus, the last layer doesn't have to do feature extraction by itself. Fine-tuning of transfer learning is done by supplanting the last layers of the network. This happens by moving the weights of the pretrained network (PTN) from source to objective dataset through alternating the nature of the *Softmax* layer.

They also used a regularization function in order to fix overfitting problems. Image data is augmented and various system structures are employed to help. One of these structures are dropout layers (fig.2.2) so that hidden weights are expelled stochastically. Another supporting method is the *early stop*, which ends training in case overfitting is detected.

Maximization of the Acc of DL algorithms are done through a function, called the *objective* function. Most known such technique is *Gradient Descent*, categorizing a model's constraints $\theta \in R^d$ by revising it in the reverse direction of the objective function $\nabla_{\theta} J(\theta)$ to parameters as η (learning rate).

$$u_t = \gamma u_{t-1} + \eta \nabla_{\theta} J(\theta)$$

$$\theta = \theta - u_t$$

Another proposed learning technique is *RMSprop* which is prioritized when the learning rate thins down drastically. It makes use of an exponentially decaying average to dispose of history so that it can get converged quickly once a convex bowl is found.

$$E|g^2|_t = \gamma E|g^2|_{t-1} + (1 - \gamma)g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E|g^2|_t + \epsilon}} g_t$$

where $E|g^2|_t$ describes the running average ($g_t = \nabla_{\theta} J(\theta)$).

Last technique that we mentioned before is *Adam*, an optimizer that calculated the adaptive learning rates of every parameter.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)g_t$$

$$u_t = \beta_2 u_{t-1} + (1 - \beta_2)g_t^2$$

m_t describes the mean of the gradients and u_t the variance of them.

The metrics chosen by the researchers are Acc, Spec, Sens, Prec, Error Rate, F1 Score and the newly mentioned Matthews correlation coefficient (MCC), which evaluates the superiority of classification as:

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

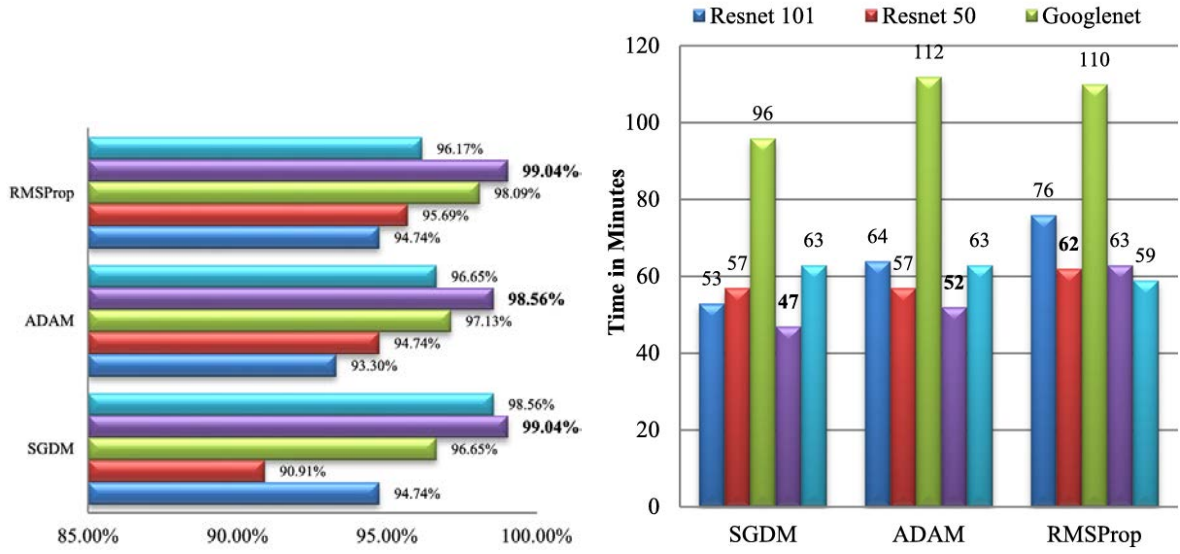
The framework used for testing uses the 255×255 MRI samples on the MATLAB R2018a platform.

Moving to results, the detailed parameters set for each implementation can be found on the tables on the original paper while all the training results are on fig.3.38. Identification of each sample is done through finetuned *Softmax* layers which are not self-trained, making an optimization hierarchy necessary. Looking on the graph of figure 3.39(α'), first observation we make is that an all optimizers, the AlexNet architecture has the upper hand, with all the rest exchanging positions, depending on the choice of the optimization process. For example, GoogleNet seems to have a balanced performance no matter the optimizer, while ResNet50 has a variance of 5%. Another interesting comparison graph that the authors provided is the comparison of training times that each technique took on every optimizer (figure 3.39(β')), as what we observe easily is that even though GoogleNet need the most time by a significant amount, it doesn't manage to reach the Acc AlexNet has with half of the amount spent on training the model. The best overall performance they computed was performed by AlexNet with augmented image sets from either SGDM or RMSProp, both reaching Acc of 99.04% with RMSProp having the inferior training time with 20 minutes more needed to reach the threshold set. Most time needed overall for training is 112 minutes, a result taken from the GoogleNet networks using ADAM. Comparing the three optimizers, the highest average time taken comes from RMSProp with 74 minutes and lowest is 63 minutes with SGDM optimization.

Training results of pretrained DL network.

Algorithms	Optimizer	Accuracy	Benign Acc	Malignant Acc	Training time
ResNet 101	SGDM	94.74%	96.70%	94.00%	52 m 40 s
ResNet 50		90.91%	96.20%	89.20%	57 m 10 s
GoogleNet		96.65%	90.50%	100%	95 m 31 s
AlexNet		99.04%	98.50%	99.30%	46 m 31 s
SqueezeNet		98.56%	95.70%	100%	62 m 42 s
ResNet 101	ADAM	93.30%	98.20%	91.60%	64 m 20 s
ResNet 50		94.74%	86.80%	99.20%	57 m 18 s
GoogleNet		97.13%	93.00%	99.30%	111 m 58 s
AlexNet		98.56%	97.10%	99.30%	51 m 31 s
SqueezeNet		96.65%	95.50%	97.20%	63 m 5 s
ResNet 101	RMSProp	94.74%	86.80%	99.20%	75 m 50 s
ResNet 50		95.69%	90.30%	98.50%	61 m 50 s
GoogleNet		98.09%	94.40%	100.00%	110 m 13 s
AlexNet		99.04%	98.50%	99.30%	62 m 41 s
SqueezeNet		96.17%	96.80%	95.90%	58 m 44 s

Figure 3.38: Training results of pretrained DL network. Table was taken from [19].



(α') Comparison of the training results of different pretrained networks. Figure is taken from [19]. (β') Comparison among the PTNs based on training time in minutes. Figure is taken from [19].

3.9 ACV-DHOA

The last biomedical research paper on brain tumors comes from Murthy et al. [20], published in 2021, who proposed their model for BT diagnosis. They dive into a different segmentation approach of MR images from the ones we have seen before as well as a different classification method. A quick overview can be seen in fig.3.40 but let's analyze it step by step. The data on which the model is evaluated has been taken from kaggle dataset published in March of 2021. It includes 253 different images, 155 being categorized as abnormal and the rest as regular images. To prepare the images for classification, they, first, use median filtering which makes a significant impact on noise reduction, give details on edges and preserves the already extracted details.

$$X_m^{MF}(r, s) = \text{median}X_m(r - v, s - w) \quad v, w \in F$$

In this equation, the 2D mask is denoted as F and X_m^{MF} are the filtered images.

Another employed technique is contrast enhancement. It works by equalizing the histogram of the image, thus enhancing the contrast of the surface on the grayscale domain.

$$X_m^{PP}(r, s) = \rho(X_m^{MF}) = \frac{\zeta_m}{Np}, 0 \leq m \leq (TG - 1)$$

where TG is the total amount of grayscale levels, $\rho(X_m)$ the probability distribution function, Np the total amount of pixels, ζ_m the total number of pixels with same intensity level m .

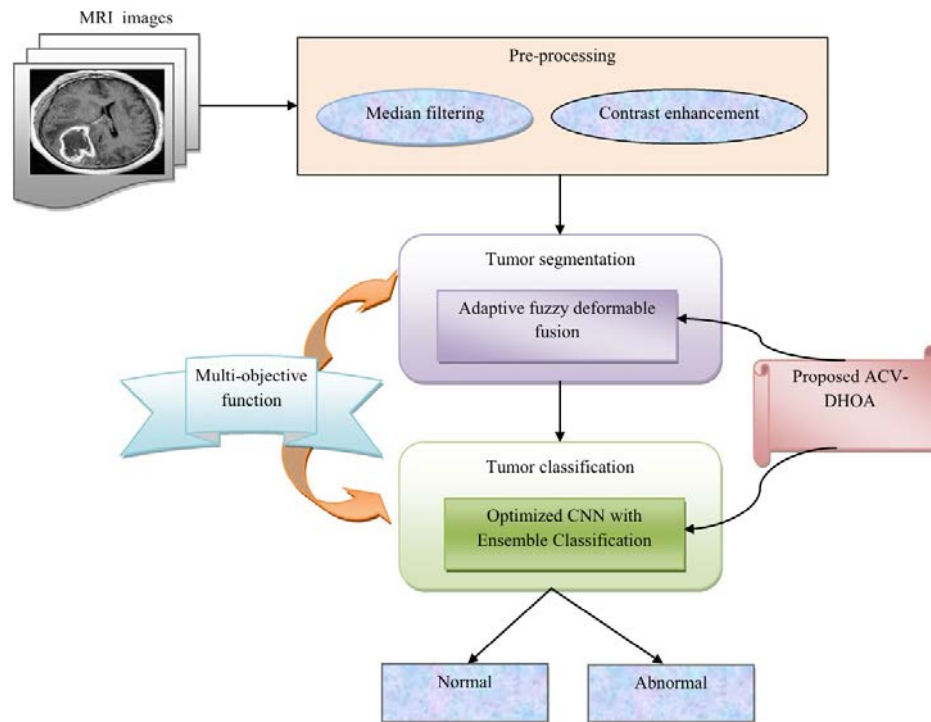


Figure 3.40: Workflow diagram of the proposed brain tumor segmentation and classification model, as designed in [20].

The diagnosis model is an approach which uses AFDF which refers to the process of segmenting a dataset, image, or signal using the Adaptive Frequency-Domain Filtering (AFDF) technique. It also uses Fuzzy C-Means (FCM), a form of unsupervised learning that assigns membership levels to each data point corresponding to clusters, allowing a data point to belong to multiple clusters with varying degrees of membership. This approach is particularly effective in medical image segmentation, where boundaries between different tissues, such as tumors and healthy tissues, are often not clearly defined. The pre-processed input are fed to FCM for clustering and later gone through snake deformation, increasing the segmentation Acc. AFDF is based on ACV-DHOA or Adaptive Cluster Validity Index based on Differential Harmony Optimization Algorithm. The key components of this method are:

- Adaptive Cluster Validity Index (ACV):
 1. Purpose: ACV is used to evaluate the quality of clustering results. It adapts based on the data characteristics to provide a more accurate measure of how well the clusters represent the underlying structure of the data.
 2. Function: It helps in determining the optimal number of clusters and ensures that the segmentation results are valid and meaningful.

- Differential Harmony Optimization Algorithm (DHOA):

1. Differential Harmony Search: This is a variant of the Harmony Search algorithm, inspired by the improvisation process of musicians. It combines the standard Harmony Search with Differential Evolution techniques to enhance exploration and exploitation capabilities.
2. Optimization: DHOA is used to optimize the clustering parameters and the segmentation process. It aims to find the best possible configuration that minimizes the clustering error and maximizes the validity index.

FCM improves the convergence rate, the Acc and the delay of the segmentation process. Cluster processing which is an essential part of FCM struggles with computational delay. Thus the addition of ACV-DHOA for optimizing the centroid cn . The fuzzy matrix is described as:

$$FM = \sum_{\alpha}^b \sum_{g=1}^h m f_{ag}^{au} d_{ag}$$

where g is the total number of pixels and α denotes the position of said pixel. The membership function matrix is represented as $m f_{ag}$ and d_{ag} describes the Euclidean distance. au show the fuzziness degree (≥ 1).

The distance between a cluster and the cluster center is equated as:

$$dis_{in} = ||px_{ip} - cc_g||$$

px_{ip} is the pixel and cc_g is the center cluster, which can be described as:

$$cc_g = \frac{\sum_{\alpha=1}^b m f_{[ag]}^{au} px_o}{\sum_{\alpha=1}^b m f_{ag}^{au}}$$

The membership function matrix is:

$$Y^{cl} = \frac{1}{\sum_{k=1}^w \left(\frac{cs_{ag}}{cs_{kg}} \right)^{\frac{2}{au-1}}}$$

where cs_{kg} is the Mahalanobis distance.

The *snake deformation* model is a well-known segmentation approach developed using energy constraints to isolate regions of interest in images. This method is applied in various fields, such as stereo tracking and motion tracking. It boasts features like high segmentation Acc and precise tumor detection in images. However, the model has several issues, including

sensitivity to local minima, representation of single objects, inability to determine the contour of every object, and Acc dependent on the convergence rate. To address these problems, the existing snake deformation model has been modified by optimizing the weight parameters using the ACV-DHOA approach to achieve optimal segmentation results. The parametric curve is modeled by:

$$PC(p) = [Q(p), U(p)]$$

In terms of boundaries, the parametric domain p has a value no greater than 1. A control point $cp_{bj} = (cp_{bj}^Q, cp_{bj}^U)^T$ should be described and the curve is a weight vector sum of such control points as:

$$PC(p) = \sum_{bj=1}^{Nb_{BS}} BS_{bj}(p) \cdot cp_{bj}$$

p is named as the curve parameter. The first B-spline basis function $BS_0(p)$ is formulated as:

$$BS_0(p) = \begin{cases} \frac{p^2}{2} & 0 \leq p < 1 \\ \frac{3}{4} - (p - \frac{3}{4})^2 & 1 \leq p < 2 \\ \frac{(p-3)^2}{2} & 2 \leq p < 3 \\ 0 & otherwise \end{cases}$$

When the active contours are considered as closed lines, then the function can be modified through arithmetic modular as:

$$PC(p) = \sum_{bi=bj-2}^{bj} BS_0(p - bj) \cdot cp_1 + mod(bi, Nb_{BS})$$

An energy function En_{snake}^{bi} is used for associating each active contour approach as measure below:

$$En_{snake}^{bi} = En_{int}^{bi} - En_{ext}^{bi}$$

both parameters of the formula depict energy, the external and the internal of the bi -th contour.

The internal energy prevents the discontinuity in the contour and can be described as:

$$En_{int} = \frac{1}{2} \int_0^1 [\alpha(|\frac{dQ}{dp}|^2 + |\frac{dU}{dp}|^2) + \beta(|\frac{d^2Q}{dp^2}|^2 + |\frac{d^2U}{dp^2}|^2)] dp$$

α and β indicate the weighting coefficients that control the tension and rigidity of snakes.

The external energy is:

$$En_{ext} = \int_0^1 (fn[Q(p), U(p)]) dp$$

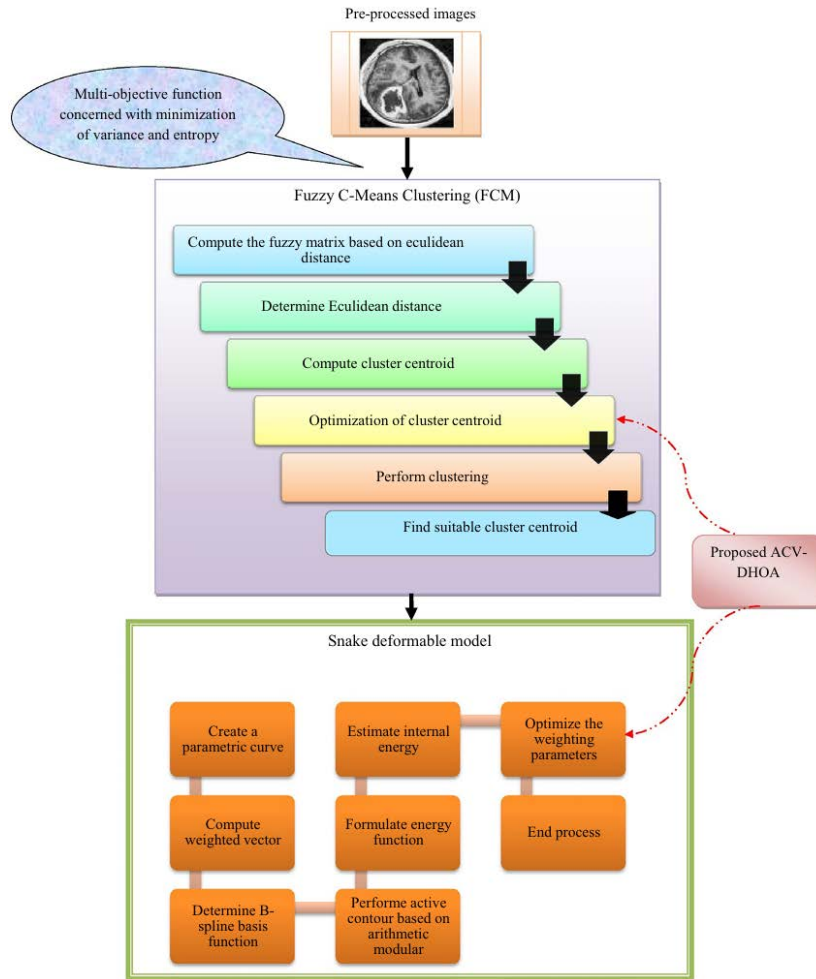


Figure 3.41: Proposed AFDF-based brain tumor segmentation model, as designed in [20].

The FCM is fused with snakes deformable model for efficient segmentation of the images, which is symbolized as X_{in}^{snake} . The scheme of this proposed segmentation approach can be seen in fig.3.41. Another point the research make clear is that their model focuses on minimizing the entropy and variance. The parameters set by them of this purpose are for the cluster centroid cn_{nk} which is between 0 and 255, where NK specifies the amount of clusters. The segmentation model goes through 50 iterations.

The classification model is also a newly proposed method named OCNN-EC. It uses the ACV-DHOA method we just mentioned to optimize the convolution layers (CL) and the number of hidden neurons (HN). In more detail, what it does is that it provides the segmented images to the OCNN. After follows feature extraction from the OCNN, then SVM, autoencoder and the DNN.

The classification of the tumor starts from the O-CNN, which controls feature learning. Then follow the rest of the architecture parts we just mentioned. The CNN has convolution layer

and pooling layer. The first one works on the heavy computations while the second subsamples the feature maps. The input of OCNN are the segmented images X_{in}^{snake} as:

$$(zs_{\phi}^{cl})_{u1,y1} = (B\alpha_{\phi}^{cl})_{u1} + \sum_{\omega_1=1}^{\psi_1\omega_1=1} \sum_{\omega_2=-\psi_1}^{\psi_1^{cl}} \sum_{\omega_3=-\psi_2}^{\psi_1^{cl}} (\zeta_{\phi,\omega_1}^{cl})_{\omega_3,\omega_2} * (zs_{\omega_1}^{cl-1})_{u1+\omega_2,y1+\omega_3}$$

where ω parameters denote the feature maps coming from every conventional filter, the bias of cl^{th} convolution layer is $(B\alpha_{\phi}^{cl})_{u1}$, $\zeta_{\phi,\omega_1}^{cl}$ denotes its weight and $(zs_{\phi}^{cl})_{u1,y1}$ its output. The activation function removes the negative values for simplicity. The output of the i -th convolution layer (activation function of $(cl - 1)^{th}$ layer) is:

$$zs_{\phi}^{cl} = \alpha f(zs_{\phi}^{cl-1})$$

The EC is conducted using a ranking approach that incorporates SVM, autoencoder, and DNN methods. Features extracted from the pooling layers are fed into three classifiers, and the highest-ranking results are deemed the optimal classification outcomes for distinguishing between normal and abnormal classes. A scheme of the final OCNN-EC classification model can be seen in fig.3.42. A step-by-step example of the model's work can be seen in fig.3.43.

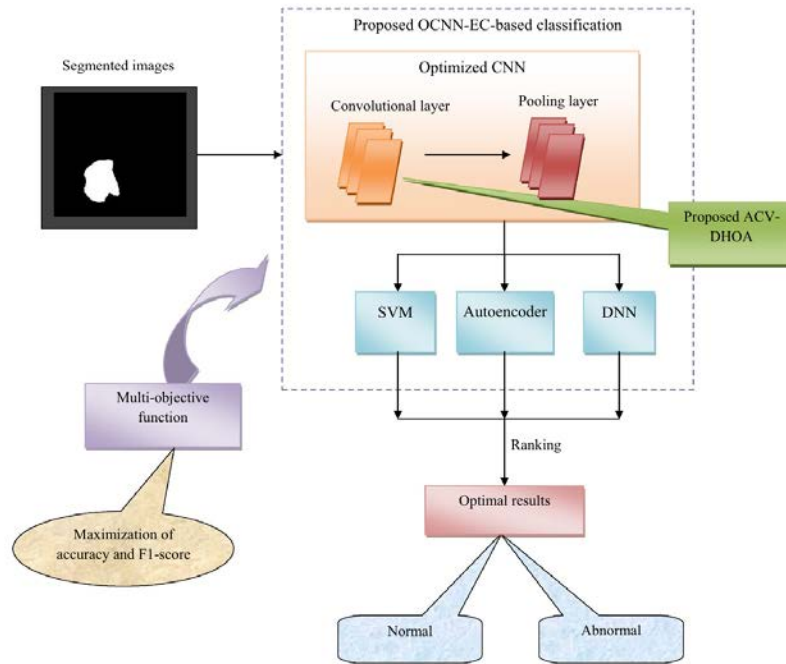


Figure 3.42: Proposed OCNN-EC-based Tumor Classification, as designed in [20].

The evaluation of the model is mainly measured through the Acc and the F1-score values. The setup of the testing is done on MATLAB R2020a on Windows 10 (64-bit) with 8GB of RAM. Some other metrics taken are:

- NPV: Probability that subjects with a negative screening test truly don't have the disease.

$$NPV = \frac{neg^{true}}{neg^{false} + neg^{true}}$$

- FDR: The number of false positives in all of the rejected hypotheses.

$$FDR = \frac{pos^{false}}{pos^{false} + pos^{true}}$$

- FNR: The proportion of positives which yield negative test outcomes with the test.

$$FNR = \frac{neg^{false}}{neg^{true} + pos^{true}}$$

- FPR: The ratio of count of false positive predictions to the entire count of negative predictions.

$$FPR = \frac{pos^{false}}{pos^{false} + neg^{true}}$$

- and Spec, Prec, Sens

The proposed ACV-DHOA-AFDF-based segmentation approach was compared with different segmentation techniques, from which we'll state a few examples the superiority of the method. The Acc of the suggested ACV-DHOA-AFDF is 0.43, 1.3, 1, 1.06, and 0.32% superior to region growing, level set segmentation, FCM, snake deformation model, and AFDF, respectively. Thus, the developed ACV-DHOA-based AFDF segmentation model increased the performance than conventional methods. On heuristic-based analysis, at 85%, the Acc of the developed ACV-DHOA-CNN-EC is 1, 1.9, 1.04 and 2.4% advanced than PSO-CNN-EC, GWO-CNN-EC, WOA-CNN-EC, and DHOA-CNN-EC, respectively.

The ACV-DHOA-AFDF is very easy to implement. The convergence rate and the performance are very high. The communication is very flexible. Because of these advantages, this method performs better than the other existing methods. The disadvantages of the existing methods are that DT is unstable and it is inaccurate often. Also, SVM needs large training time for large dataset and NN requires more data than the other methods.

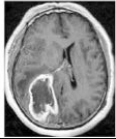
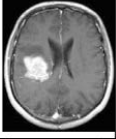
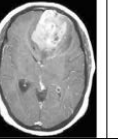
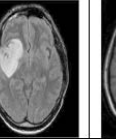
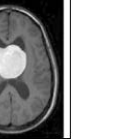
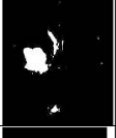
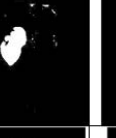
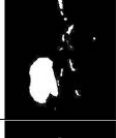
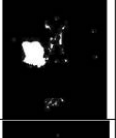
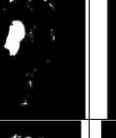
Image description	Image 1	Image 2	Image 3	Image 4	Image 5
Original images					
Segmented images using DHOA-AFDF [26]					
Segmented images using GWO-AFDF [40]					
Segmented images using PSO-AFDF [39]					
Segmented images using WOA-AFDF [41]					
Segmented images using ACV-DHOA-AFDF technique					

Figure 3.43: Sample segmented image results of the proposed brain tumor segmentation model using optimized AFDF model. Figure is taken from [20].

3.10 Overall comparison

3.10.1 Accuracy comparison

If we compare all the above methods regarding the brain tumor diagnosis, we see that the research on this field has taken important steps forward, as it not only focuses on Prec for the diagnosis of the patient but also on the important factor of the computational cost a system like that makes. It can be easily understood that a low-cost inference model would benefit greatly the medicine industry as the manufacturing of these models would not be costly, meaning that there could be a high number of systems, not particularly the same between each other so that there would also be less of a risk of a false diagnosis. In terms of performance, the 2018 model proposed from Nayak et al.[6] still has the best Acc overall (figure 3.44).

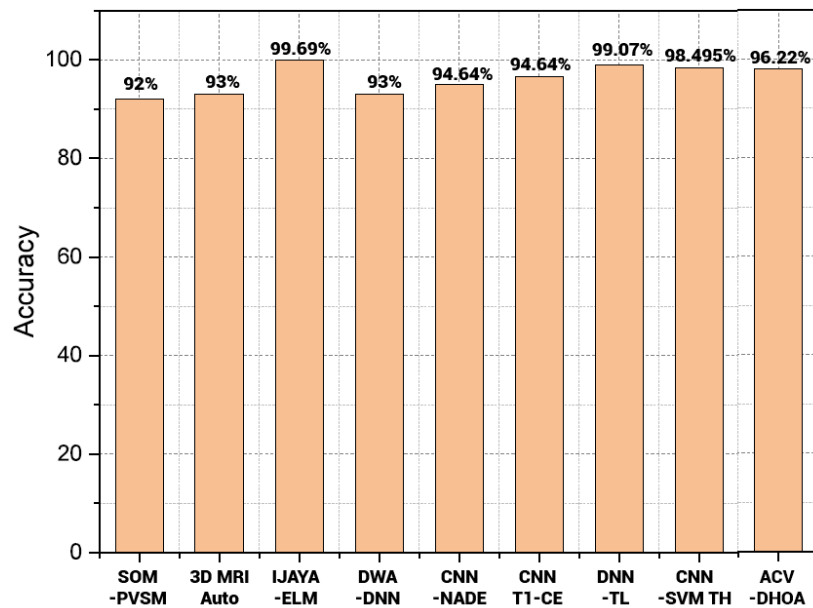


Figure 3.44: Comparative study of accuracy for different Brain Tumor detection and classification techniques.

3.10.2 Cost comparison

In order to test the BT methods fairly, we tested them on our system on the same dataset. The dataset chosen is the BraTS 2017 dataset that includes 243 scans (135 GBM and 108 LGG) from faithful sources. Our system consists of Matlab 2023a, an AMD Ryzen 5 5600X CPU, an NVidia RTX 3060 GPU and 16GB of RAM.

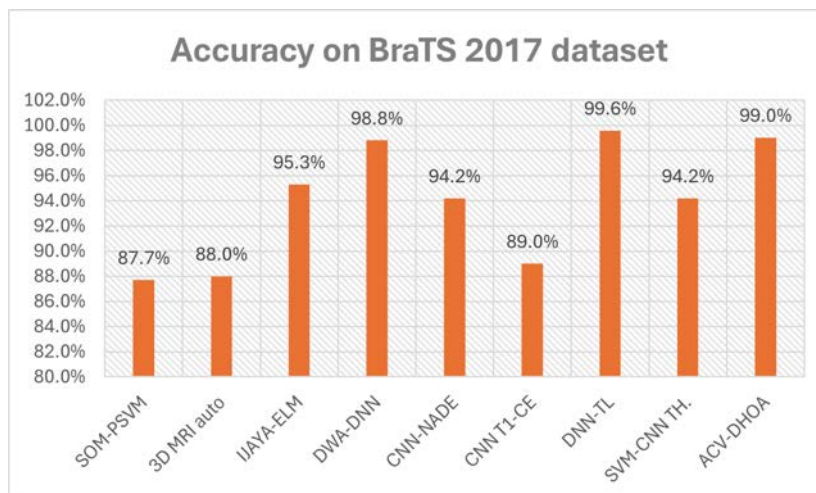


Figure 3.45: Comparative study of Acc for different Brain Tumor detection and classification techniques on the BraTS 2017 dataset.

The tests on our system (fig.3.45) operate quite differently, as the best accuracy is achieved from **DNN-TL** with **99.6%** correct classification rate. All three top spots come from the three DNN-based methods, as after there is a distinct step down in performance with a 3.5% drop. The worst performance after 50 epochs comes from **SOM-PSVM**, as it only manages to reach 87.7% Acc.

When we compare by cost, the results are quite different (fig.3.46,3.47). The methods that use DNNs (**DWA-DNN**, **DNN-TL**, **ACV-DHOA**) have higher architectural overhead by nature, as DNNs have higher complexity to chase high accuracy, a fact that is also confirmed from the Acc's reported on the previous subsection. From these, **DNN-TL** has the worst architectural design with 465668 hidden neurons and best execution time, as the TL addition increases the already big overhead but speeds up the process. Next would be the **ACV-DHOA** which even though optimizes the number of neurons with the use **OCNN-EC**, it still uses SVM and DNN. After follows **DWA-DNN** which has a high overhead while also one of the lowest Acc's. Going away from DNNs, we move to **CNN T1-CE** which has good complexity as it doesn't need to add neurons because of the efficiency of the training algorithm. After we move to **IJaya-ELM**, which has the best learning accuracy while also having a pretty compact architecture, as it only uses 4039 hidden neurons but needs the most execution time, with 1504 seconds needed for 50 epochs. Next is the **CNN-NADE**. NADE has a pretty compact design but needs at least two CNNs in order to be precise. Both **SOM-PSVM** and **SVM-CNN TH**. have comparable as their main overhead comes from the SVM interpreted in both with the threshold mechanism in the second helping a bit. An interesting fact is that SVM uses 2102 neurons but using approximation lowers this number to 1914 (10% less). The lowest architectural overhead comes from the **3D MRI Auto**, as it uses LBP as its learning method, a very simple algorithm.

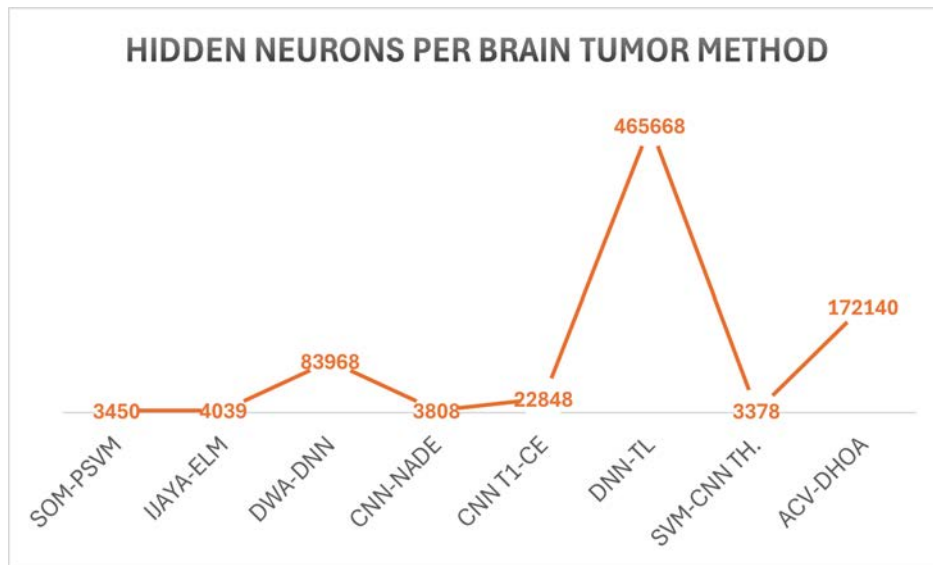


Figure 3.46: Comparative study of the number of hidden neurons for different Brain Tumor detection and classification techniques on the BraTS 2017 dataset.

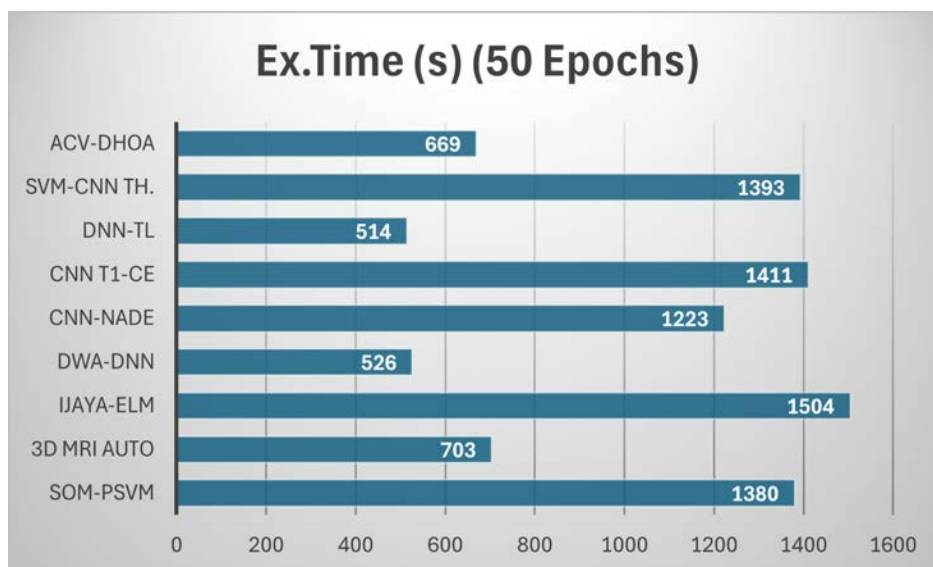


Figure 3.47: Comparative study of the number of the execution time for different Brain Tumor detection and classification techniques on the BraTS 2017 dataset.

Chapter 4

Cervical cancer

Cervical cancer occurs when health cells in the cervix develop mutations in the DNA. Generally, there are 2 types of cervical cancers: ‘Squamous Cell Carcinoma’ and ‘Adenocarcinoma’. It occurs mostly in women over the age of 30 and the main cause is HPV (Human Papillomavirus). The test checks for abnormal cells on the cervix of the patient that are cancerous or have the potential to become cancerous and may also detect certain infections and inflammation. It is detected using a pap test and HPV-DNA test. If this cancer is detected in advance through regular screening, it gives a lot of insights to the physicians to provide better treatment [21].

4.1 SVM Cerv

The first take on the research of the cervical cancer AI-based diagnosis came on 2004 from Zhang and Liu[22], Robotics Institute of Carnegie Mellon University. The way this method works is similar to what we have seen before, as it is based on feature extraction from imaging samples. This time, the input are multispectral images also known as PAP smear images which they analyze while in the process they are trying to remove the redundant features found on the samples.

First step of this implementation is the segmentation of the image to take away the background of the image, as this information takes away some Acc of the end outcome. Normalizing all band images is next by subtracting the spectral signature of the aforementioned background. Next follows feature extraction. In this step, the process is to take each pixel and extract the information included in it in order to get a universal idea. The information that collected are:

- Pixel intensity
- Daubechies 2 and Daubechies 16 asymmetric orthogonal wavelets
- Biorthogonal wavelet
- Gabor wavelet

The multidimensionality of the multispectral images come from applying the feature extraction process on every band.

The feature set collected gets pruned by the SVM model that uses the screening algorithm to remove irrelevant data, as at the end of step 2 there are near 4000 features. The decision function can be written as:

$$h(x) = w \cdot \Phi(x) + b = \sum_{i=1}^n a_i y_i K(x, x_i) + b$$

x_i depicts the training sample and y_i its class label. A transformation function ($\Phi(\cdot)$) maps the data points into a higher dimensional feature space ($D > d$). A kernel function ($K(\cdot, \cdot)$) is employed to calculate the inner product of mapping. The a parameters are optimized by finding the hyperplane in feature space with maximum distance to the closest image ($\Phi(x_i)$) from the training set, which reduces to solving a linearly constrained convex quadratic program.

We also get a surface point s from the equation:

$$h(s) = h(az_1 + (1 - a)z_2) = 0$$

where z_1, z_2 are two points such that $h(z_1)h(z_2) < 0$.

The unit normal vector $N(s)$ at this boundary point is given by:

$$N(s) = \nabla h(s) / \|\nabla h(s)\|$$

$$\nabla h(s) = \frac{\partial h(s)}{\partial s} = \sum_{i=1}^n a_i y_i \frac{\partial K(s, x_i)}{\partial s}.$$

$N(s)$ indicates the orientation in the input space where the projected training data are distinctly separated in the vicinity of s . Consequently, the difference in orientation between $N(s)$ and any direction u can be utilized to assess the local discriminative relevance of that direction at s . The difference can be found as $|u^T N(s)|$. Last but not least, the decision boundary scatter matrix can be computed as:

$$M = \int_B N(s) N^T(s) p(s) ds$$

In case the number of samples is already known (l), we can replace the previous formula with:

$$\hat{M} = \sum i = 1^l \hat{N}(\hat{s}_i) \hat{N} \frac{(\hat{s}_i)^T}{l}$$

This value can be expanded to the multi-dimensional space that the problem is implemented on. It is measured either in pairs or one-vs-all fashion.

The data are first getting pruned away from the decision boundary in order to reduce the computational cost and negate the effect of outliers have on the error of the outcome. Then follows the one-vs-all SVM to solve Q-class problems. A concern of this method is the complexity of the algorithm. That is regulated by several parameters as the number of samples (l), the number of boundary points and the error margin (ϵ) but the researchers also note that none of the above had a serious effect on the matter. Another important observation made is that if a feature (u) has the characteristic that: $u^T M u = 0$, it means that it has no relevant information for the context but the inherent uncertainty doesn't allow this estimation, as an uneven distribution can mask an almost same Acc.

Next section of the model is filtering. The input of the filtering sequence is the feature vector we just saw. Then each class is run through a quadratic discriminant:

$$g(x) = \sum_{i=1}^k \frac{1}{\lambda_i} [\phi_i^T (x - \mu)]^2 + \sum_{i=k+1}^n \frac{1}{\beta^2} [\phi_i^T (x - \mu)]^2 + \ln[\beta^{2(n-k)} \prod_{i=1}^k \lambda_i]$$

where:

- μ : class mean
- λ_i, ϕ_i : the i -th eigenvalue and eigenvector of covariance matrix Σ
- $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$
- β : positive constant
- Also, the discriminative filter output is $h(x) = g_{normal}(x) - g_{cancer}(x)$

After filtering has been applied, the detection of malignant data is relatively easy. An example can be seen in fig.4.1.

1. Smooth $H = h(x)$ with a Gaussian filter.
2. Find all local maxima m_i in H in their respective regions (R_i) and define the point of it that have values above a fixed fraction ($h(m_i) \geq 0.5$).

3. For each R_i follows the extraction of a geometric feature $G = C/L$, where C is the circumference of the region and L is the distance from the local maxima to the boundary. These R_i are pruned only if $h(m_i) < 0.5 \cdot \max_i h(m_i)$ or $G < 2$. After checking all R , the final candidate region set is built.
4. The candidate regions that have overlapping areas are merged for simplicity.

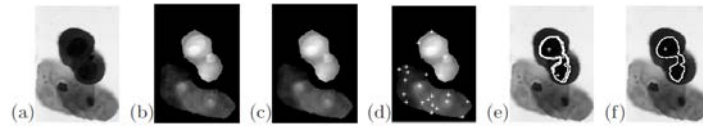


Figure 4.1: An example of cancerous region detection taken from [22]. (a) Original image. (b) Scaled output surface from discriminative filtering. (c) Gaussian smoothing of (b). (d) Local maxima points found in (c). (e) Contours of candidate cancerous regions. (f) Merged result.

The evaluation of the model was done on 40 PAP images with each having 52 spectral bands on 149 cells, of which 41 are cancerous and 108 are regular. The image sizes go from 93×64 to 300×227 . For feature extraction they chose a 16×16 block in each band, which extracts a total of 156,732 sample vectors (26,064 positive, 132,063 negative).

They compare the proposed method to IG and AVR. In more detail, for each type of wavelet, images were randomly into training set (32 images) and test set (8 images) for a number of times. The recorded FPR per subset dimension can be seen in fig.4.3. The parameters before and after the SVM screening are also in fig.4.2. The selected 68 features of the original full sample set are used for evaluation purposes with the use of the modified quadratic discrimination. From the comparison graph, it's easy to understand that the SVM outperforms the competition, especially in high TPR values. From other tests, they also extracted the information that decreasing the feature space lower than 40 doesn't significantly reduce the Acc. From the 41 pictures, 40 of them were classified correctly, bringing the Acc to a 98% for a small dataset.

	DB2	DB16	Bio2.2	Gabor	Combined
Original Dim.	800	800	900	1200	3700
After Screening	48	42	52	30	68

Figure 4.2: Various dimensions before and after SVM based feature screening. Values taken from [22].

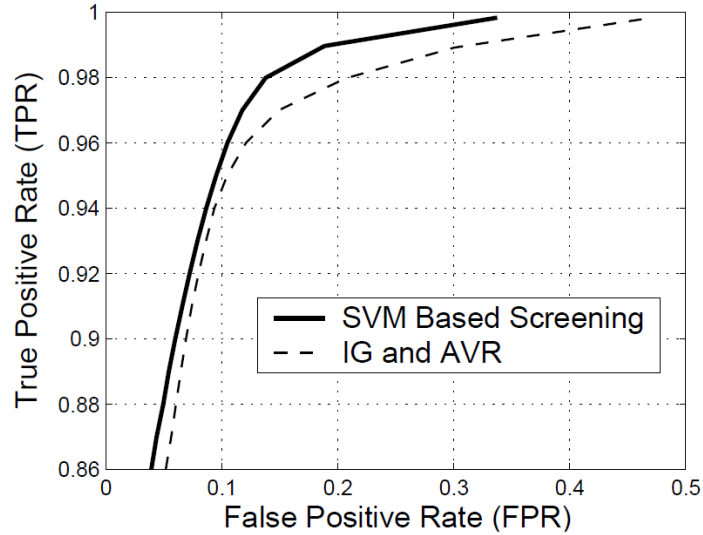


Figure 4.3: Comparison between SVM and IG+AVR based screenings, as recorded for the SVM Cerv [22].

4.2 CNN-superpixel

In 2014, Youiyi Song et al.[23] proposed a superpixel and CNN based segmentation method that takes advantage of the contrast a cytoplasm has to the rest of the area. Let's discuss on detail how this architecture is built.

An overview of the proposed framework can be seen in figure 4.4. Other informational figures are 4.5 and 4.6 that show the detailed segmentation procedure.

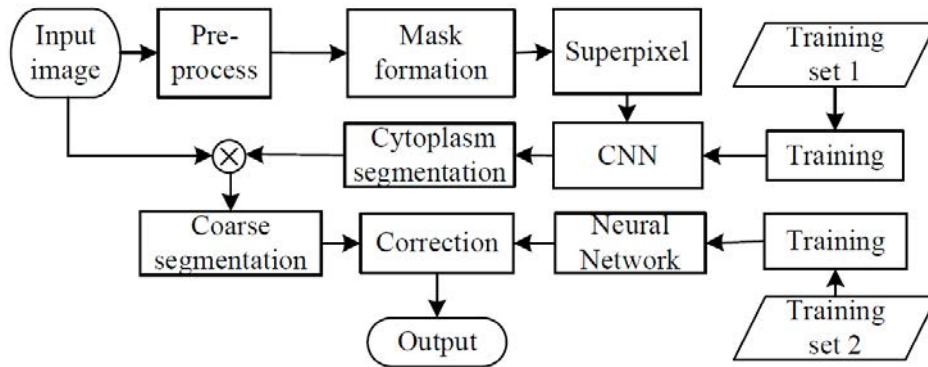


Figure 4.4: Framework of the CNN-superpixel method, as proposed in [23].

On the pre-processing step, filtering reduces the noise of the image. Let's consider C_k the color value of the $m \times m$ window. A trimmed mean filter gets applied:

$$C = \frac{1}{(1 - 2\lambda)m^2} \sum_{k=\lambda m^2+1}^{(1-\lambda)m^2} C_k$$

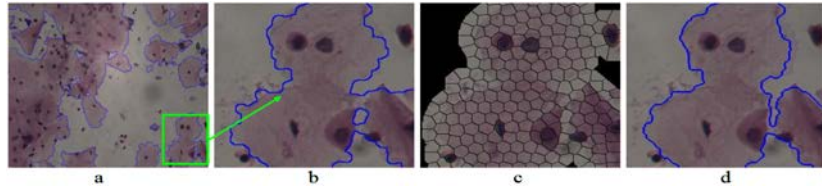


Figure 4.5: Illustration of cytoplasm segmentation. (a) Cytoplasm boundary after coarse segmentation. (b) Image of the area in the green frame. (c) Superpixel extraction from figure 4.4. (d) Cytoplasm boundary after fine segmentation. [23]

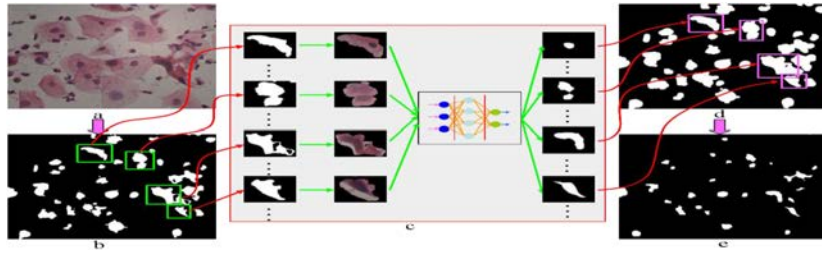


Figure 4.6: Illustration of segmentation; a. Input image b. Coarse segmentation, c. Segmentation correction; d. Fine segmentation, e. Final segmentation results. [23]

where the researchers chose $\lambda = 0.4$, $m = 11$ as the optimal parameters.

Next is masking to segment the input coarsely and reduce the delay introduced by the lack of identifying the significant regions. The nucleus region is stained in red while the rest don't have such characteristics. For the exact region, the research team decided to transform from RGB display of colors to CIE LAB to increase this contrast. Scaling up the gray levels of the image from $[r_{min}, r_{max}]$ to $[0, 255]$ is done by:

$$f(x) = \frac{D_m}{2} \left\{ 1 + \frac{1}{\sin(\alpha \frac{\pi}{2})} \sin \left[\alpha \pi \left(\frac{x}{D_m} - \frac{1}{2} \right) \right] \right\}, 0 < \alpha < 1$$

where D_m is the maximum value after transformation, x is gray pixel value, α is a weight parameter.

Because of the several variance parameters a cervical cancer cell can show, a global threshold value is not suitable, thus the researchers employed the high-dimensional Otsu thresholding method. There are also morphological alternations to smoothen the boundaries and eliminate impurities.

After the masking process, the sample is left with uneven staining and bad illumination that would lead to poor performance, especially around the edge. In order to refine the quality, they introduced the superpixel algorithm, a technique already found in cases as blood vessels, optic disc glaucoma and prostate segmentation. With the resource-limited nature of the

systems built, they chose the simple linear iterative clustering (SLIC) algorithm with five dimensional features (CIE LAB of L, A, B and two dimensional coordinates) to improve the superpixels' shape. After coarse segmentation, around 15 pixels wide are reserved to split the coarse image.

Features as size, color shape and texture have to be extracted for each pixel for the segmentation to be considered successful. The color information is saved in 18 individual color features that with their correlation affect the classification process. Spatial information are explored to characterize data as the appearance and shape of each superpixel and its neighborhood. On the images that come for analyzing, erroneous nucleus segmentation may occur due to cytoplasmic staining or overlapping of inflammatory cells. Thus, RGV converts to HSV, using the V channel for this procedure. Additionally, morphological top-hat transformation is performed to enhance the contrast of the image. Also, they developed a new nucleus template for same purposes and better image binarization. Let's consider the pixel value as $L(x, y)$ and the Gaussian mask as $G_\sigma(x, y)$. Then, the method could be described as:

$$T(x, y) = L(x, y) * G_\sigma(x, y) - b$$

The operations done on this part also manage to smoothen the border of the nucleus region while also managing to not go out of it in the process, due to the threshold value.

Last procedure is some refining of the values for imbalance purposes with cross-validation with a magnitude of 5 pixels, as the segmentation does a great job on splitting the area. The result is the final nucleus, the outcome from the updated nucleus with point-based pixel value in each test.

The database that the researchers decided to focus on has 200 patients, aged from 22 up to 64 from the Sixth People's Hospital of Shenzhen. 53 slides were collected via the Olympus BX43 microscope at a magnification rate of 40. From the total of 21 images, 15 were found with abnormalities while 6 contained no harmful spots. All slides were prepared using manual liquid-based cytology (MLBC) technique stained with H&E, a simple and cheap method accepted from the medical field for evaluation which is also confirmed by biopsy in the end. The whole amount cell samples are 1400 from background data. CNN uses around 85% for training and the rest for testing. The image size is 1240×1360 , with the total procedure of segmentation taking around 50s. The system has 3.39GB of RAM available on an Intel i5-3470 CPU, with clocks set to 3.20GHz.

The axes around which they evaluate the performance of the model are BPNN, SVM and

LVQ algorithms. The metrics are Acc, sensitivity and specificity. Looking at figure 4.7 we see that the proposed CNN outperforms the competition, confirming how effective DL is for nucleus and cytoplasm detection. Focusing more on the performance of the segmentation

Algorithms	Accuracy	Sensitivity	Specificity	F1 measure
BPNN	0.8900	0.8578	0.9286	0.8947
PNN	0.8775	0.8912	0.8647	0.8753
SVM	0.8975	0.8841	0.9119	0.8993
LVQ	0.9000	0.9124	0.8883	0.8985
CNN	0.9450	0.9406	0.9495	0.9453

Figure 4.7: Nucleus region detection results per NN of a number of different metrics. [23]

process, this time we utilize only F1 measure, Prec and Rec. The automatic method seems more preferred by the medical community. The results can be seen in fig.4.8. As observed, the proposed segmentation shows great potential, as it achieves the best performance amongst other popular techniques. Visually, we can also see the difference in fig.4.9. We can see that the segmentation method developed works even under dyed shades, uneven illumination, dust blocking or abnormal nuclei.

Method	F1 measure	Precision	Recall
Canny	0.5488±0.0215	0.9084±0.0008	0.4094±0.0202
Otsu	0.5622±0.0202	0.5073±0.0292	0.6837±0.0284
ALGC [2]	0.8732±0.0015	0.8497±0.0047	0.9016±0.0004
Our method	0.8951±0.0215	0.9143±0.0202	0.8726±0.0008

Figure 4.8: Nucleus region segmentation results per method on CNN-superpixel, using mean value and standard deviation. [23]

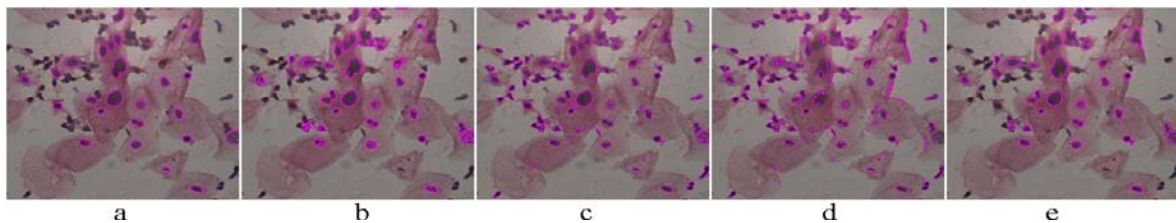


Figure 4.9: Comparison of segmentation methods found in CNN-superpixel [23]. (a) Ground truth. (b) Canny method. (c) Otsu method. (d) Method in [24]. (e) The CNN-superpixel method.

4.3 DeepPap

In 2018, M. Summers et al.[25] published their own on early detection of cervical cancer. In the training phase, a ConvNet is first pre-trained on the ImageNet dataset. On the pre-processing of the model, the researchers decided to go out of the norm and not operate directly on images containing full cells for practical reasons, as the segmentation of individual cells is a challenging problem. TBS rules [26] state that different cervical cytology abnormalities are associated with different nucleus abnormalities. Hence, nucleus features in themselves already include substantial discriminative information. The images are constructed are $m \times n$ patches centered around the nucleus centroid. The method that they chose for classification uses data augmentation by applying N_r rotations on each cell image, resulting to an increased number of samples which might have slightly worse quality on high frequency data due to the rotation. From these come the training data, which originate from a patch from the center of each augmented sample. The significant increase of samples (even though with lower quality) is crucial to the success of ConvNet. Also, the Herlev and HEMLBC dataset (fig.4.10) has a very limited number of images, so the augmented images have even bigger role. The developers also avoid the use of padding for simplicity reasons.

The prediction of the nucleus center might not be completely accurate, so we translate up to d random pixels of each nucleus centroid N_t times, obtaining N_t points, shaping a "centroid area". These areas produce $N_t m \times n$ sized patches and are used as training samples. An example of this behavior can be seen figure 4.11. With that technique, the samples for training are increased and create a simulation environment for inaccurate nucleus center detection. The images found in the Herlev dataset have around 3 times more abnormal cell images than regular data. Thus, there's the need to also alternate the proportions of positive and negative samples, since classifiers tend to prefer the majority class which are the abnormal cells on this case. This, also, improve the Acc and convergence rate of ConvNets in the training phase.

Category	Class	Cell type	Num.
Normal	1	Superficial squamous epithelial	74
Normal	2	Intermediate squamous epithelial	70
Normal	3	Columnar epithelial	98
Abnormal	4	Mild squamous non-keratinizing dysplasia	182
Abnormal	5	Moderate squamous non-keratinizing dysplasia	146
Abnormal	6	Severe squamous non-keratinizing dysplasia	197
Abnormal	7	Squamous cell carcinoma in situ intermediate	150

Figure 4.10: The 917 cells (242 normal and 675 abnormal) from Herlev dataset.

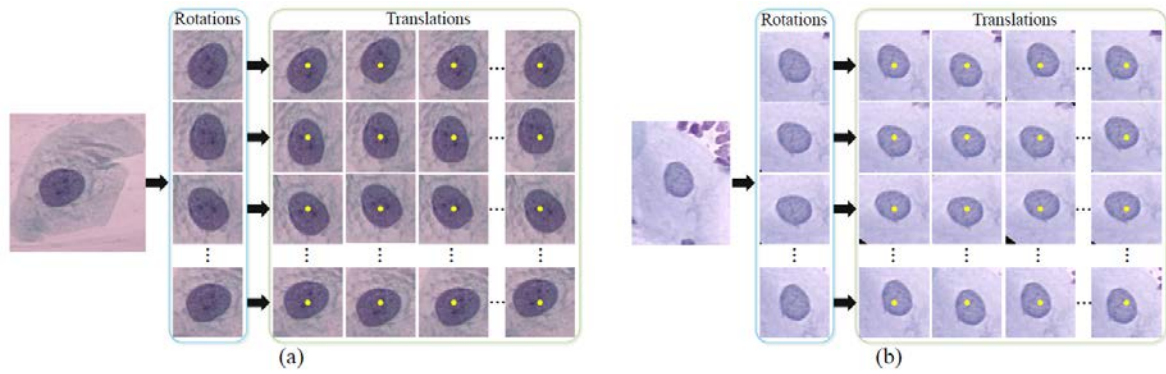


Figure 4.11: Two set of image patches are generated from (a) an abnormal and (b) a normal cervical cell image by rotations and translations. The centroids of translated image patches are shown as yellow, indicating the potentially inaccurate detection of nucleus centers. [25]

The ConvNet model is a DL model that includes various stages, as:

- 2D convolutional layer
- non-linear layer with ReLU classifier and max-pooling layers
- more convolutional neurons
- fully-connected layers with included dropout to reduce overfitting

As input, it uses the image that is the output of the raw pixel intensity subtracted by the mean value. The output layer has as many neurons as the classes extracted. There is, also, weight optimization, as the classification error gets minimized by backpropagation. Initially, these values are updated with the gradients of the loss function, computed via stochastic gradient descent (SGD) over a mini-batch (size of 256) of training samples. The finalization of training is triggered after 30 epochs.

The research team have also interpreted transfer learning in their method, a fine-tuning technique that uses a few of the convolutional and pooling layer of the ConvNet model. It takes inspiration from the BVLC CaffeNet network [27], as it also takes RGB input arrays. In the current case, transferred layers get fine-tuned with a learning rate 10 times smaller to the original CaffeNet value.

Looking at the datasets closely, the Herlev data come from the University Hospital of Herlev which has 917 images with resolution of $0.201 \mu m$ per pixel, each containing an cervical cell (fig.4.12(α')). In total, there are seven discrete classes, which are categorized as: (1-3):

normal and (4-7): abnormal. For each of the last classes, we execute 10 rotations of 36 degrees and 10 translations. In the other hand, on normal cell we execute 20 rotations of 18 degrees and 14 translations. In the end, there are 100 and 280 image patches for each image, respectively. This fixes the distribution on the training phase. The size per patch is set to 128 pixels, covering most of the full nucleus region for every case. They are then, upscaled to a $256 \times 256 \times 3$ pixels via bilinear interpolation due to transfer learning. From this patch, we select a 227×227 randomly selected subsection from which we subtract the mean image values.

On the HEMLBC dataset, it's information collected from the People's Hospital of Nanshan District by using another model that Summers and his team had developed before on autofocus (fig.4.12(β')). The specimen go through manual liquid-based cytology with H&E staining. They are in total 989 abnormal cells from 8 CIN slides that have been confirmed through biopsy and 1381 normal cells from another 8 NILM slides, from which we randomly select 989 cells for balancing properties. Most of them are segmented by an automated algorithm [24]. The normal cells are split into two subsets: the first collected by a pathologist with automated segmentation and the second is some false positive cells collected during bootstrap process from validation images with manual segmentation for the ill-segmented ones by an engineer. On both normal and abnormal images, we execute 10 rotations of 36 degrees with 10 translations, creating 100 patches per cell image.

The ConvNet learning rates are set as:

- $conv_1 - pool_5 = 0.001$
- $fullyconnected_6 - fullyconnected_8 = 0.01$
- After each ten epochs, there's a decrease by 10 on both.
- Weigh decay = 0.0005
- Momentum = 0.9
- Dropout ratio ($fullyconnected_6, fullyconnected_7$) = 0.5

It runs on the Caffe platform, using an NVIDIA GTX TITAN Z GPU with 12GB of RAM. For evaluation, they used as metrics sensitivity, specificity, Acc, H-mean, F-Meas, area under the ROC curve.

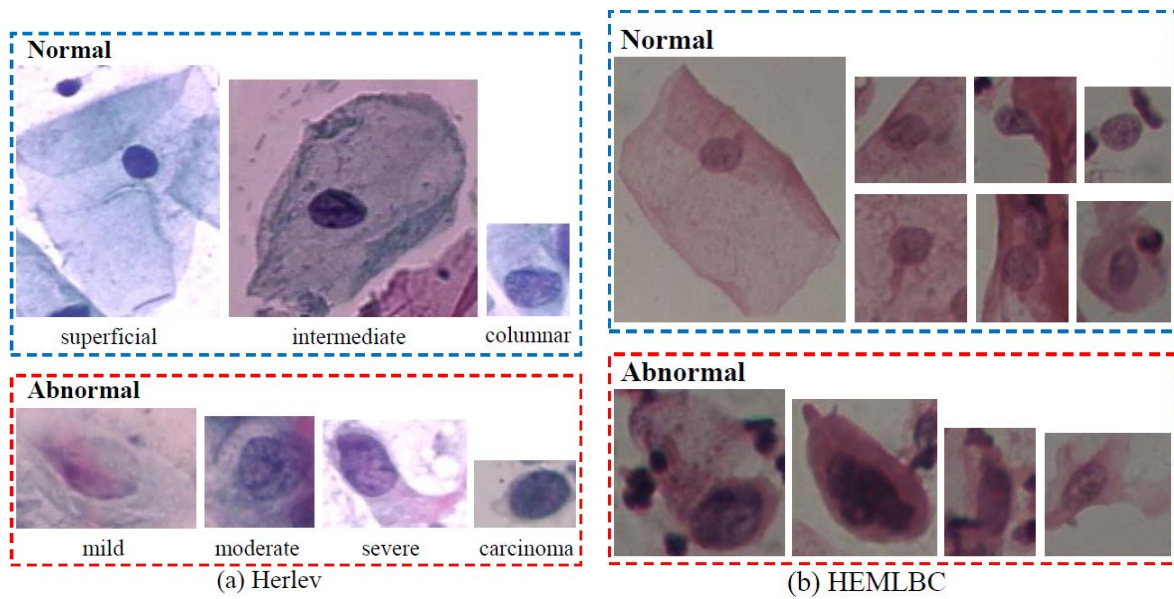


Figure 4.12: Example images of normal and abnormal cervical cells from the (a) Herlev and (b) HEMLBC dataset. All these examples keep their originally relative scales for better illustrating the different characteristics (mainly nucleus size) between normal and abnormal cells.

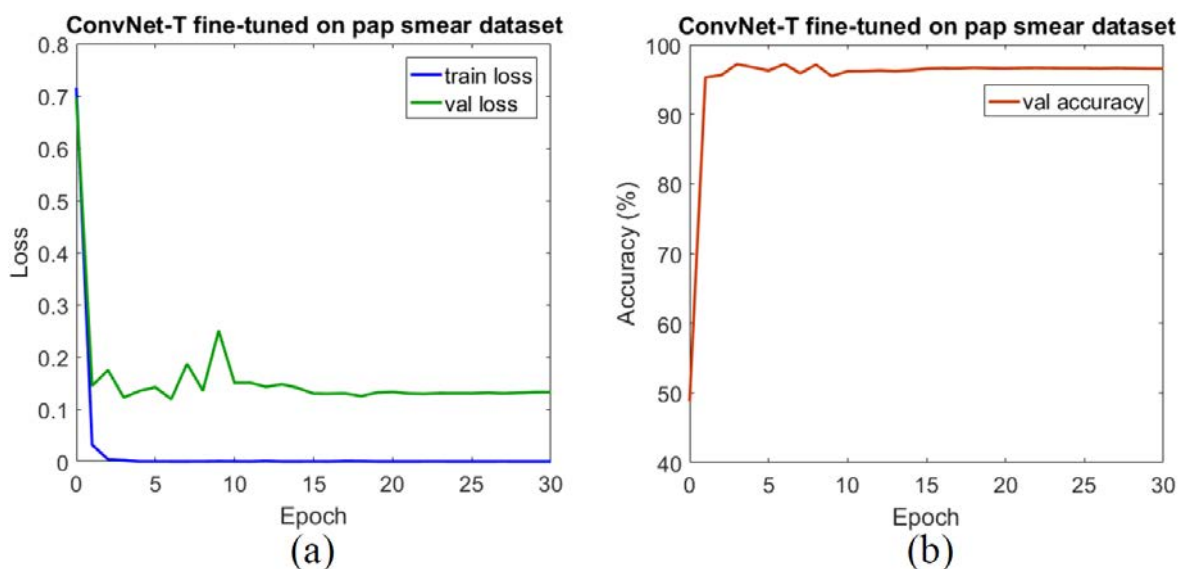


Figure 4.13: (a) Training and validation loss, and (b) validation accuracy versus number of training epochs on DeepPap method.

On figure 4.13 we see how the ConvNet-T performs on the Herlev dataset. After reaching 6 epochs, we find the minimum value of the validation loss value (0.119) with a validation Acc

of 0.972. Figure 4.14 shows the learned filters of the first convolutional layer of ConvNet-T trained on the Herlev PAP smear dataset. These automatically learned filters mainly consist of gradients of different frequencies and orientations and blobs of color, which are necessary for the cervical cell classification task.

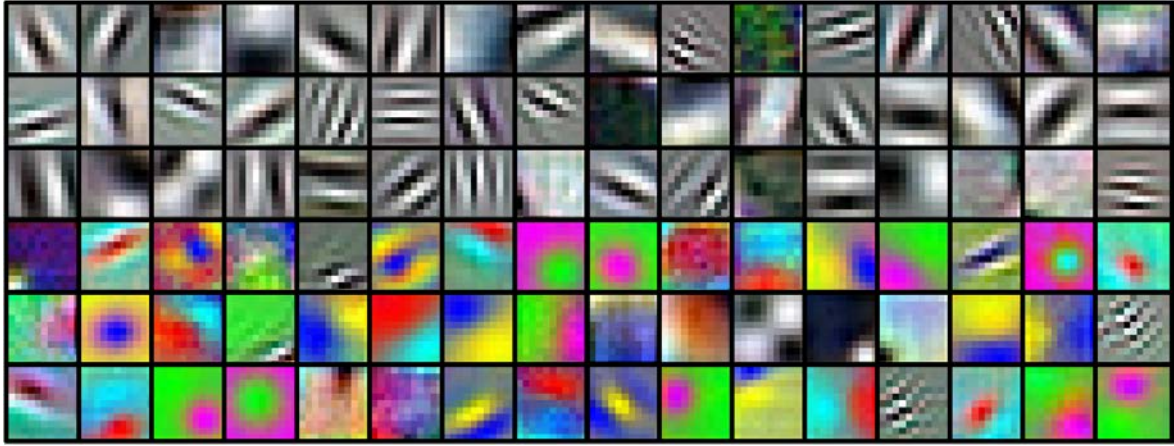


Figure 4.14: Visualization of the 96 filters with size of $11 \times 11 \times 3$ in the first convolutional layer of DeepPap fine-tuned on Pap smear dataset.

Looking on fig.4.15, we see that the mean values that we get for the proposed method are: **Sens:** 98.2%, **Spec:** 98.3%, **Acc:** 98.3%, **H-mean:** 98.3%, **F-meas:** 98.8%

Comparing with the rest of the mentioned methods, we see that there are better results in every aspect, apart from sensitivity, which it's slightly lower.

Methods	k-fold CV	Sens (%)	Spec (%)	Acc (%)	H-mean (%)	F-measure	AUC
Benchmark	10	98.8 $\pm$ 1.3	79.3 $\pm$ 6.3	93.6 $\pm$ 1.9	(88.0 $\pm$ NA)	-	-
PSO-Inn	5	98.4 $\pm$ NA	92.2 $\pm$ NA	96.7 $\pm$ NA	(95.2 $\pm$ NA)	-	-
GEN-Inn	5	98.5 $\pm$ NA	92.1 $\pm$ NA	96.8 $\pm$ NA	(95.2 $\pm$ NA)	-	-
ANN	LOO	99.9 $\pm$ NA	96.5 $\pm$ NA	99.3 $\pm$ NA	(98.2 $\pm$ NA)	-	-
K-PCA + SVM	10	-	-	-	96.9 $\pm$ NA	-	-
Ensemble	5	99.0 $\pm$ NA	89.7 $\pm$ NA	96.5 $\pm$ NA	-	93.1 $\pm$ NA	-
ENS	5	-	-	-	-	-	0.884
dis(S + M)	5	-	-	-	-	-	0.964
ConvNet-T	5	98.2 $\pm$ 1.2	98.3 $\pm$ 0.9	98.3 $\pm$ 0.7	98.3 $\pm$ 0.3	98.8 $\pm$ 0.5	0.998
ConvNet-T Δ^5	5	98.4 $\pm$ 1.3	97.1 $\pm$ 2.3	98.0 $\pm$ 0.8	97.7 $\pm$ 1.0	98.7 $\pm$ 0.6	0.994
ConvNet-T Δ^{10}	5	98.2 $\pm$ 1.2	96.7 $\pm$ 3.1	97.8 $\pm$ 0.5	97.4 $\pm$ 1.2	98.5 $\pm$ 0.4	0.994

Figure 4.15: Performance comparison of the CNN of DeepPap method with previous methods on the Herlev dataset.

The table of fig.4.16 shows the number of accurate classification for each of the seven cell classes. The proposed method shows perfect Acc on two types of normal cell as well as one abnormal type. The abnormality scores that we see are close to 0 and 1 respectively for most normal and abnormal cells. The threshold value of 0.5 highly helps with that. The 7-class problem has an overall error of 1.6%, a great result compared to previous iterations.

Cell type	Correct classification
Superficial squamous epithelial	74 (100%)
Intermediate squamous epithelial	70 (100%)
Columnar epithelial	94 (95.9%)
Mild squamous non-keratinizing dysplasia	182 (100%)
Moderate squamous non-keratinizing dysplasia	145 (99.3%)
Severe squamous non-keratinizing dysplasia	189 (95.9%)
Squamous cell carcinoma in situ intermediate	147 (98.0%)

Figure 4.16: Numbers (and percentages) of correct classification (normal vs abnormal) of DeepPap method for each of the seven cell classes in Herlev dataset.

Looking on the output from the tests on the HEMLBC dataset, the team also compares to the MLPs that have been tried on the current dataset before (fig.4.17).

Methods	<i>k</i> -fold CV	<i>Sens</i> (%)	<i>Spec</i> (%)	<i>Acc</i> (%)
MLP	5	92.5±NA	96.0±NA	94.3±NA
ConvNet-T*	5	98.3±0.7	99.0±1.0	98.6±0.3

Figure 4.17: Performance comparison of CNN of DeepPap with a previous method on the HEMLBC dataset.

The computational speed of the ConvNet-T was around 4 hours for 30 epochs, or 3.5 seconds per cervical cell during testing.

In general, what makes this method more preferable compared to the previous mentioned methods is, at first, the robustness of it to inaccurate detection of nucleus centroid. Also, the architecture shows great potential even in difficult environments, where there's no uniformity on chromatin distribution or the nuclei is much larger. At last, the high sensitivity and specificity of the model on testing on both dataset means that it is able to work great in PAP smear and H&E stained liquid-based cytology. Such performance leaves a lot of potential for automation-assisted systems in primary cervical cancer screening.

4.4 AlexNet-SVM

In 2017, Taha et al.[28] proposed their own take on how to tackle the fifth deadliest cancer for women. It is based on a DL approach that uses pap-smear images. They take an already developed CNN for feature extraction that is followed from an SVM classifier.

The CNN takes the position of handmade and customized features. In terms of the architecture, all the basic operation parts are present, as convolution layers, subsampling, followed by the SVM which takes the place of the fully connected neurons. Training a CNN can be

tricky in this field, as we need large dataset for that which is not the easiest thing to find. To tackle this problem, the developers opted for the economic alternative of a pre-trained CNN, which saves a great amount of time and needs only fine-tuning for the pre-trained weights. The chosen CNN from the team is AlexNet, a NN we have seen before. It has been trained with 1.2 million images with 1000 different classes.

After extracting the features from the fully connected layers, next follows the classifier which is an ANN, in this case a SVM, which is chosen for the high-dimensionality prowess which allows mapping a huge number of feature vectors across different layers. Also, to tackle over-fitting, SVM is assumed to be less sensitive to that because of its characteristics we already mentioned.

For the SVM training phase, each feature vector is labeled as norm or abnormal from the values -1 or 1, respectively, creating a "feature-class" pair of $\{x, y\}$. If we consider L such pairs and i an iterator with boundaries of $[1, L]$, and y the output ($y \in 1, -1$) and D the vector size. The hyper-plane of the two classes is:

$$w^T x + b = 0$$

Separating the classes using the above, they built the equation below:

$$\min(\frac{1}{2}w^T w + C \sum_{i=1}^L \xi_i)$$

where C is known as the penalty parameter while also, we fulfill the constrain $y_i(w^T x_i + b) \geq 1 - \xi_i$, $\xi_i \geq 0$ for $i = 1, \dots, L$. The optimal hyper-plane, allows a classification to be done according to a decision function as:

$$f(x) = \text{sgn}(w^T x + b)$$

The evaluation of the model is done on the Herlev dataset we also used on the previous model. The CNN achieves good quality in terms of the classification power required. AlexNet model was trained on the ImageNet dataset which doesn't have medical imaging samples. This introduces the problem that we have to know which layer performs the best, as the feature extraction is essential for the overall performance. The only layers that are considered by the rest of the model are the deep layers, starting from Conv4. Due to the transfer method chose, the output feature layers are frozen while the features from the output act as training

samples for the SVM classifier. If, let's say, the fifth convolution layer is the output layer, this means that the weights between the first and fifth have pre-trained values, which leaves only the SVM values to be acquired. While this might seem great in theory, in reality there is still left a great amount of weights to be computed, as the dimension of the obtained feature vector, for example, from the Conv4 is $13 \times 13 \times 384$ which comes up to 64896 features. 70% of the data are accommodated for training while the rest for testing. Taking a look at the table of fig.4.18(α'), we see overall, the best performance comes out from a deep layer, in this case *fc7*, which are more descriptive than their shallow layer counterparts. There's also another set of tests where the columnar class is still present, setting the normal cell images to 242. The results can be seen in fig.4.18(β').

Experiment 1				
		Recall	Precision	Accuracy
CNN feature	Conv4	99.01%	99.02%	98.37%
	Conv5	100%	98.54%	98.78%
	FC6	99.5%	99.01%	98.6%
	FC7	99.51%	99.5%	99.19%
	FC8	98.6%	97.8%	97.9%

(α') The recall and precision values for 2-class classification of pap-smear without the columnar class.

Experiment 2			
		Recall	Precision
CNN feature	Conv4	94.1%	87.6%
	Conv5	97.04%	89.14%
	FC6	99.01%	85.2%
	FC7	64.04%	96.3%
	FC8	59.11%	95.24%

(β') The recall and precision values for 2-class classification of pap-smear including the columnar class.

Overall, the numbers that recorded for the performance of the model are: **Rec**: 99.51%, **Prec**: 99.5%, **Spec**: 97.67%, **Acc**: 99.19%.

These number show how advanced the method already is on correctly diagnosing abnormalities. Future references would suggest a more resource-focused revision, as the power that the current shows comes with its boundaries.

4.5 SSAE-SC

Next year D.Selvathi and A. Aarthi Poornila gave a try on the cause, proposing a 2-level based image classification using PAP-smear images [29]. The distinct steps of the system can be seen in fig.4.19.

On the pre-processing of the samples, the motivation is the same as we have seen be-

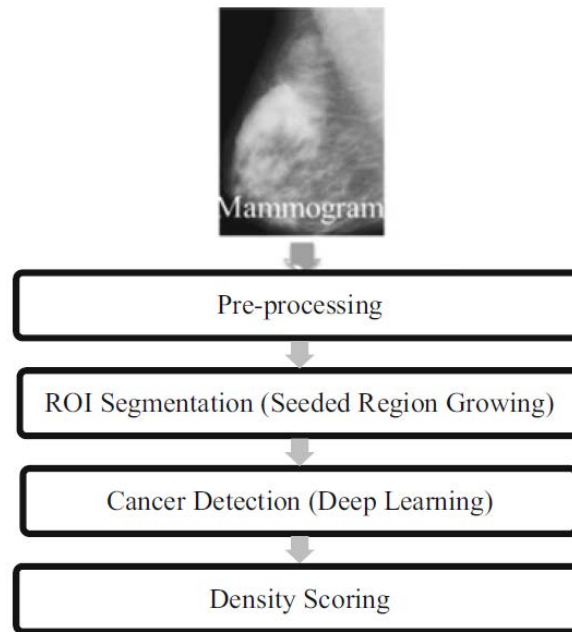


Figure 4.19: Flow chart of SSAE-SC system.

fore: feature extraction and suppress noise. Mammogram has critical problems with noise constrains that can alternate heavily the Acc of the model. To tackle this, a median filter is applied. Another way to extract unwanted information is by multiplying the original image with a binary mask. It's a favourable technique in pectoral muscle removal, which involves the initial seed point selection and determination of the neighboring pixels.

Next follows a DNN for supervised learning of features and a sparse autoencoder (SAE) as well as a stacked SAE (SSAE) for unemployed feature training. On parameter learning, while an individual layer is not deep, the architecture itself is. First is the extraction of sub-patches from randomly selected parts of the image. Then, these sections are encoded. Last is the transformation to a feature, using a *Softmax* classifier. The two groups of the final classification are normal and risky mammograms which means that they have at least one match dense in useful information.

The model is developed in MATLAB R2016a on Windows 10 and evaluated on an Intel Core i7-5500 CPU clocked at 2.40GHz with 8GB of RAM.

The 2D median filter is of size 3×3 . Next follows the artifact suppression and background separation. The object is read as a binary image. After checking the amount of pixels per area, the largest one gets selected and using morphological opening operation, all other binary information is removed except the breast profile. After follows the removal the outliers by setting the pixel's value to 1 if and only if there are 5 or more 1's in its 3×3 neighborhood.

This results in the final image where each element is of radius 5, flat and disk-shaped. Then follows segmentation from the background by multiplying the binary information with the original, grayscale mammogram image.

The pectoral muscle has to be taken out from the breast profile. This is done through seeded region growing (SRG) which has some distinct steps:

1. Initialization

- *Seed Selection*: Select initial seed points that are representative of the region of interest (in this case, the pectoral muscle). These seed points can be chosen manually by a user or automatically based on certain criteria such as intensity or texture.
- *Parameter Setting*: Define criteria for region growing, such as intensity thresholds, similarity measures, or HOM criteria.

2. Region Growing

- *Neighbor Analysis*: Examine neighboring pixels of the current seed points to determine if they should be added to the region. This is done by checking if they meet the predefined criteria set in the initialization step.
- *Pixel Addition*: Add neighboring pixels that meet the criteria to the region, effectively expanding the region.

3. Region Merging

- *Merge Criteria*: Combine adjacent regions that meet specific criteria to form larger homogeneous regions. This step helps in refining the segmentation by ensuring that similar regions are not segmented separately.

4. Iteration

- *Repeat Process*: Continue the process of region growing and merging iteratively. Each iteration involves analyzing new neighboring pixels and adding them to the region if they meet the criteria.
- *Convergence Check*: Check for convergence, i.e., when no more pixels can be added to the region based on the criteria. This indicates that the segmentation process is complete.

5. Post-processing

- *Refinement*: Perform additional processing steps to refine the segmented regions. This can include smoothing boundaries, removing noise, or filling small holes within the segmented region.
- *Validation*: Validate the segmentation results to ensure Acc. This can be done through visual inspection or by comparing the results with ground truth data if available.

In the context of pectoral muscle removal from mammograms, these steps help in accurately identifying and isolating the pectoral muscle region, which can then be excluded from further analysis to focus on the breast tissue.

Training of the DNN is split in two parts: supervised and unsupervised learning. Supervised learning is a compromise in terms of computation time, as it adds considerable delay but is very important for the Acc of the CNN. In the other hand, both SAE and SSAE are trained in an unsupervised way but the Softmax classifier that follows the whole network is trained as usual.

The database chosen by the developers is the mini-MIAS mammographic set, which contains 322 images of 1024×1024 size.

In fig.4.20 we see the effect of de-noising the image has. Next follows the binary image which in combination with the noise-free equivalent, create the final, pre-processed image (fig.4.21).

The DNN takes the images, splits them into 100×100 segments for complexity purposes. There are specific proportions in which the patches are then selected: 10% are background patches, 45% contain fat tissue and the rest 45% contain dense tissue. The learning rate set for the CNN is 0.001 for optimal results. A simple comparison can be seen in fig.4.23. In terms of decoder function, the best performance comes from linear transfer. Also, the loss function chosen is cross-entropy. In terms of training algorithms, we get comparable results, as five of the tested algorithm get the same Acc of 98.5%. These are the Levenberg-Marquardt, BFGS Quasi-Newton, Resilient Backpropagation, SCG and CG with Powell/Beale restarts. Amongst them, the one chosen is the backpropagation, as it takes the least time to reach that performance. Another experiment set by the researchers was changing the number of samples used for training. As expected, the Acc climbs from 62.7% with 50 training images to 98.5%

with 270 samples. In fig.4.22 we see how the removal of the pectoral muscle is done.

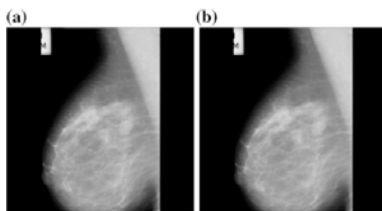


Figure 4.20: An example of a) an original and b) the denoised counterpart of it using the SSAE-SC method [29].

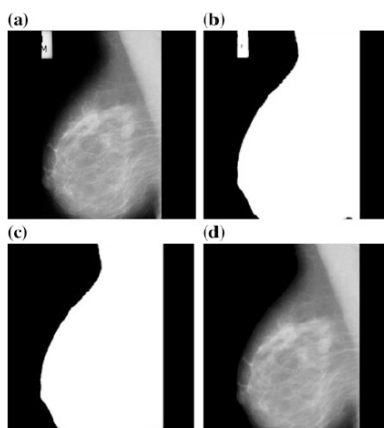


Figure 4.21: An example of a) the denoised image, b) the binary counterpart, c) morphologically opened image and d) the final preprocessed image by using the SSAE-SC method [29].

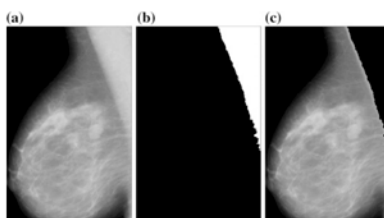


Figure 4.22: An example of a) the preprocessed image, b) segmented pectoral muscle and c) after pectoral muscle removal, using the SSAE-SC method [29].

In fig.4.24 we see a direct comparison between the CNN, SAE and SSAE. The last seems to have the upper hand in all cases.

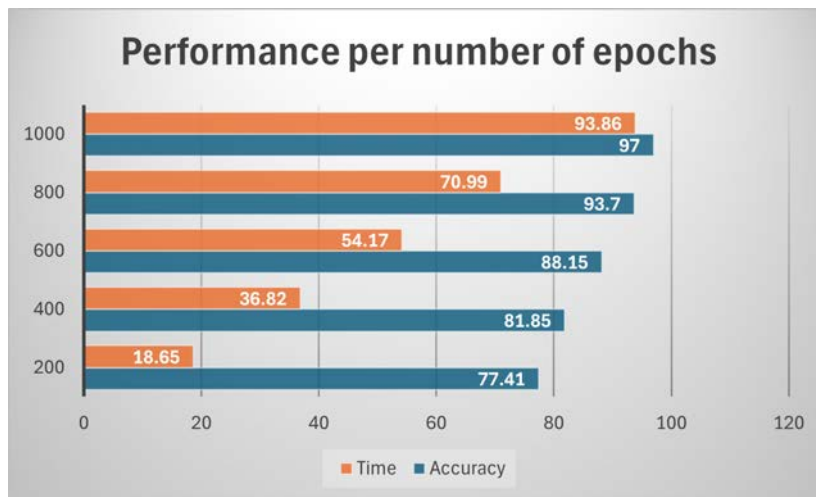


Figure 4.23: Comparison per number of epochs, using the SSAE-SC method [29].



Figure 4.24: Comparison of performance of the three DL steps of SSAE-SC method [29] using MIAS database.

It is certain that even if one of the patches extracted is classified as dense, the mammogram is labelled as abnormal. Otherwise, a normal mammogram has only fatty or background patches.

4.6 CNN-SVM

In 2020, Dongyao Jia et al. [30] made a new framework based on features retrieved by GLCM and a Gabor filter are co-existing with the features extracted from a CNN. They are, then, give to the SVM to achieve classification. Let's take a closer look to learn more about the architecture (fig.4.25).

On pre-processing, various methods are used to increase the amount of data available. First is rotation is which is done by 90-degree steps, resulting in a dataset four times greater already. In terms of scaling, we double and half the initial size of images, resulting to three times the initial dataset. Next is random cropping, which helps with the stability of the model. When applied, the mode is described as *RA1*, otherwise *RA0*. Last pre-processing method mentioned is shear-warp transformation, which changes both axes as explained below:

$$X_{dir} = \begin{bmatrix} 1 & -\tan\alpha \\ 0 & 1 \end{bmatrix}, Y_{dir} = \begin{bmatrix} 1 & 0 \\ -\tan\alpha & 1 \end{bmatrix}$$

What we understand considering the nearest-neighbor interpolation is that the target pixel connects directly with its neighborhood. The four modes of the transformation are: a) horizontal (SH1), b) vertical (SH2), c) 60 degrees (SH3) and d) the original which is labeled as *SH0*.

To extract the features correctly, all factors that can affect the performance of the model have to be taken into consideration. Texture, morphology and chroma are some of those that if not handled carefully, they can deteriorate the end result by far.

First technique used for feature extraction is GLCM, a method we are already familiar with from a previous model. The advantages we take from it is its simplicity, freeing up the need of obvious regularity. Next is Fourier transform, which helps with texture patterns, a matrix of 2D fourier transformation matrix.

Gabor transformation, on the other hand, doesn't do well with locality. The Gabor function is:

$$\phi(x, y, w, \theta) = \frac{1}{2\pi\sigma^2} \exp\left(-\left(\frac{x'^2 y'^2}{2\sigma^2}\right)\right) [\exp(iwx') - \exp\left(-\frac{w^2\sigma^2}{2}\right)]$$

where:

$$x' = x\cos\theta + y\sin\theta$$

$$y' = y\cos\theta + x\sin\theta$$

$$w = 2\pi * f$$

We understand that the handling parameters of the transform are the frequency scale f and the angle θ set. In this paper, θ is selected as $0, \frac{\pi}{8}, \frac{\pi}{4}, \frac{3\pi}{8}, \frac{\pi}{2}, \frac{5\pi}{8}, \frac{3\pi}{4}, \frac{7\pi}{8}$, and f as $0, 1, 2, 3, 4$, adding up to 40 texture features, 40 filters. Last features extraction method found is the Markov random field.

LeNet-5 model's hidden layer is responsible for feature extraction. As we see in fig.4.26, the

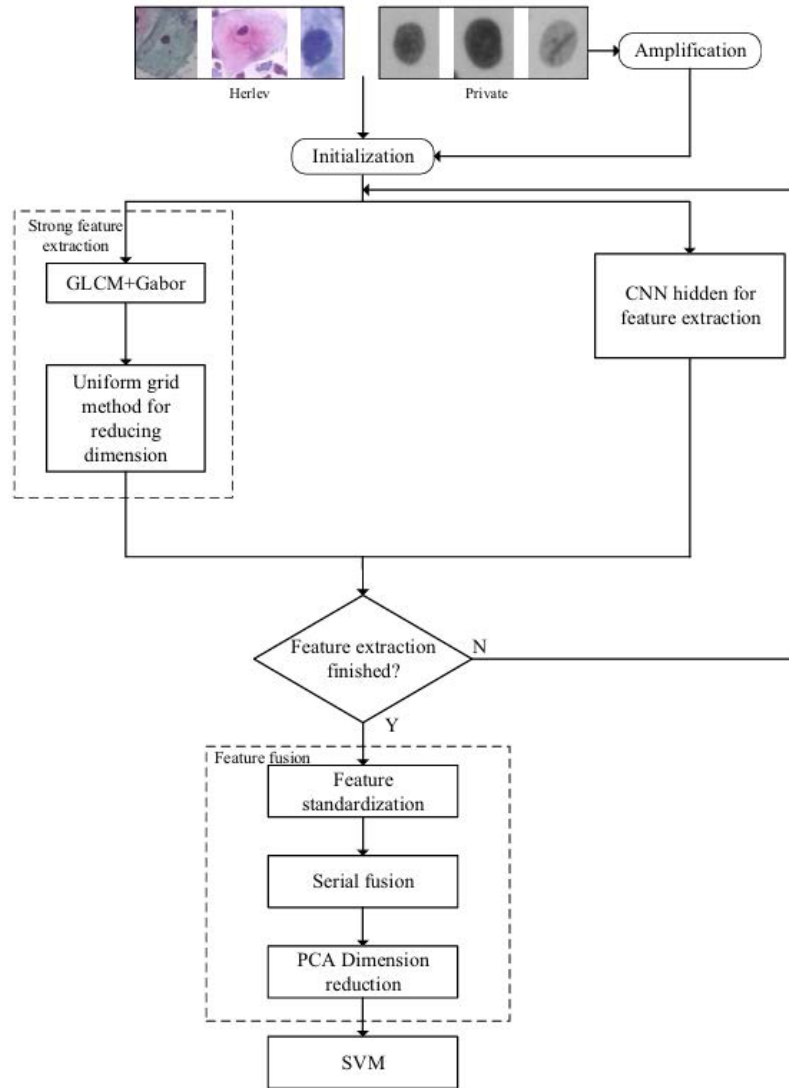


Figure 4.25: Flow chart of the proposed classification algorithm CNN-SVM, given in [30].

input layer takes a 70×70 image of epithelial cells. C1 layer consists of five cores, each one building its own feature map with size of $(70-5+1) \times (70-5+1) = 66 \times 66$. S2 downsamples the previous vertices via max-pooling. 2×2 non-overlapping areas are selected from the initial graph. From these, we select the maximum value which correspond to the unique value of the area. The feature maps on this layer have size of $(66/2) \times (66/2) = 33 \times 33$. C3 layer is constructed by setting a visual field window of 5×5 with step size equal to 1 and 12 feature maps. The graph obtained is $(33-5+1) \times (33-5+1) = 29 \times 29$. The last layer (S4) is a max-pooling instance with 2×2 window with step size of 2. The 12 feature maps of S4 layer with size of $(29+1)/2 = 15$, bringing the final feature dimension to $15 \times 15 \times 12 = 2700$. The developers also note that the activation function for every layer is the sigmoid function.

They also use feature fusion to reduce the complexity. The process is as follows:

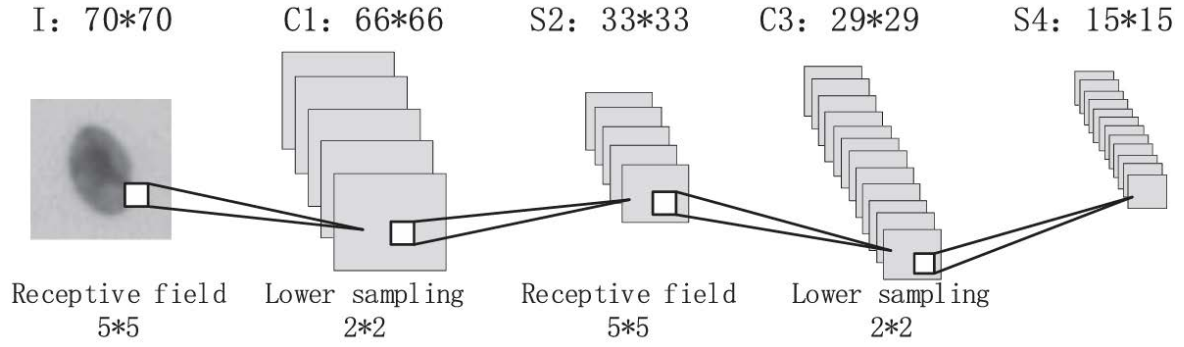


Figure 4.26: Proposed LeNet-5 model in [30] for feature extraction.

1. Standardization of parameters: Considering μ as the mean and σ the standard deviation of the eigenvector parameters, the expected value is computed as:

$$x'_i = \frac{x_i - \mu}{\sigma}$$

2. Multi feature serial fusion: Next is the Gabor + GLCM feature vector which is symbolized as $X_1 = [x_{11}, x_{12}, \dots, x_{1m}]$ and the abstract feature set as $X_2 = [x_{21}, x_{22}, \dots, x_{2n}]$. The combination of the two can be seen as $W = (w_1, w_2, \dots, w_{m+n}) = (\alpha X_1, \beta X_2)$, where α, β are two parameters, each with a range of 0 to 1 and the rule that $\alpha + \beta = 1$. They mention that from testing, it seems that the optimal values are $\alpha = 0.4, \beta = 0.6$.
3. Feature dimensionality reduction: After fusing the data, the vector elements come up to $2700 + 2560 = 5260$. To lower the complexity, they proposed to construct an overall scatter matrix (P) for the set of training samples is shown as follows:

$$P = \frac{1}{N} \sum_{i=1}^N (a_i - m)(a_i - m)^T = AA^T$$

where a_i = represents the training samples, m is the mean vector of sample A and N is the number of samples. Then, PCA is applied, taking the first 799 features, making a characteristic matrix $W_{799 \times N}$.

The architecture is developed on Python 2.7 and tested on an Intel Core i8 CPU with clock up to 2.8GHz and 16GB memory.

There are two datasets for testing. First is the Herlev dataset we mentioned on the previous models tested. The other is a private dataset of 2000 cell images that come from 200 different cases from screenings between August 2016 and December 2017 in Guangdong Province

People's Hospital. From the dataset, half of it were diagnosed from the medical staff as positive. 80% of the data were used for training with fusion ratio of 2:8. 5-fold cross-validation is used in all data blocks. In the table of figure we see the parameter settings of model A-I. RO depicts rotation, SA is scaling, RA is random crop and SH is shear-warp transformation. The rotation seems like it helps the Acc and should be considered in the future. On mode A&B&C, mode SC is a sensitive change while scaling mode SC0-SC1 improves Acc while on the other hand SC0-SC2 doesn't. An important note left from the developers is that scaling can improve performance and robustness but on if used with caution, as model C shows the other side of the coin. Comparing A&D to C&E, random crop improves the Acc of the model. On mode A&F&G&H, model SH doesn't seem to work so it won't be applied on future references. So, in conclusion, model J was chosen as the best model as its Acc of 91.1% was the best.

Rotation, scaling and random-crop transform to the data follow next, resulting to magnification times of $4 \times 3 \times 2 = 24$ while keeping the minimum at 1. We see the performance achieved in figure 4.27. As datasets grow, the parameters follow so with worse performance achieved when the magnification is set to three.

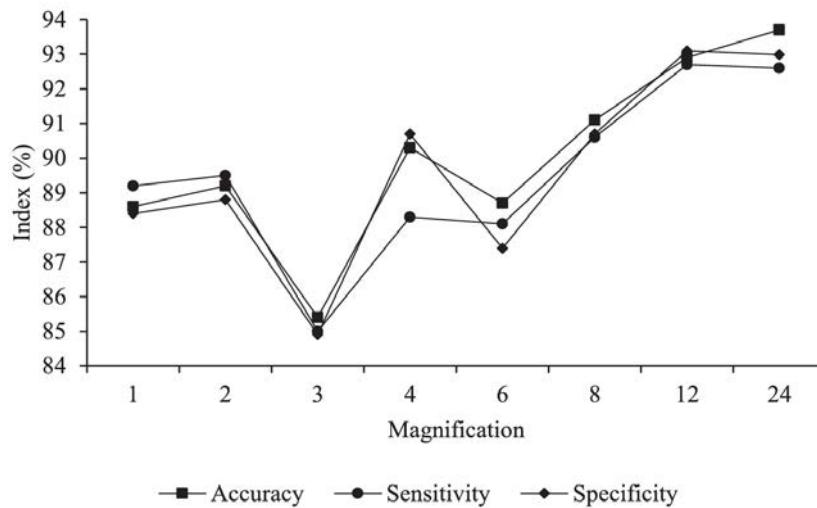


Figure 4.27: Comparison of model accuracy under different scale datasets from the values recorded in [30].

Next we investigate how different feature extraction methods interact with the model, having as input normalized samples. The methods tested are GLCM, Fourier, Gabor, Markov random field and their combinations. Feature vectors were fed to the SVM classifier. The results can be seen in figure 4.28. What we get is that the combination of GLCM with Gabor

has the best performance both in Acc and processing time. Even though MRF shows potential, it considers only a small area near the point rather than other areas. Since the data of the private dataset show continuity in their content, the performance of MRF can be understood as artificial. Also, the general performance is better when the samples are left in small sizes.

Next we search to find the sweet-spot for the combination of the weights from the strong feature neurons (α) and the CNN (β). The data shown in figure 4.29 sets clear that the more we balance the two parameters, the better the classification. The best classifier is when $\alpha = 4$, $\beta = 6$.

Accuracy and running time of feature extraction methods on private dataset.

Method	70 × 70		48 × 48	
	Accuracy	Time (s)	Accuracy	Time (s)
GLCM	88.6	31	89.1	27
Fourier	89.8	46	89.5	41
Gabor	91.3	53	90.1	48
MRF	75.7	38	76.5	35
GLCM + Fourier	87.0	48	84.4	45
GLCM + Gabor	92.1	46	90.9	49
GLCM + MRF	89.5	58	87.3	42
Fourier + Gabor	91.4	45	90.7	43
Fourier + MRF	85.9	44	82.8	49
Gabor + MRF	82.6	50	79.3	45

Accuracy of classifiers under different ratios of α, β .

Weight Number	[1,9]	[2,8]	[3,7]	[4,6]	[5,5]	[6,4]	[7,3]	[8,2]	[9,1]
1	0.899	0.912	0.938	0.939	0.932	0.912	0.909	0.799	0.681
2	0.907	0.918	0.928	0.932	0.922	0.913	0.901	0.782	0.672
3	0.899	0.916	0.928	0.926	0.921	0.922	0.898	0.799	0.692
4	0.903	0.932	0.933	0.934	0.929	0.921	0.902	0.801	0.601
5	0.902	0.920	0.939	0.948	0.939	0.913	0.901	0.803	0.703
6	0.903	0.913	0.925	0.930	0.921	0.920	0.910	0.798	0.718
7	0.901	0.913	0.932	0.938	0.931	0.901	0.903	0.795	0.693
8	0.902	0.916	0.920	0.921	0.921	0.913	0.902	0.803	0.702
9	0.890	0.911	0.931	0.934	0.924	0.911	0.898	0.801	0.691
10	0.906	0.910	0.921	0.932	0.925	0.909	0.902	0.799	0.679
Mean value	0.901	0.916	0.930	0.933	0.927	0.914	0.903	0.798	0.683

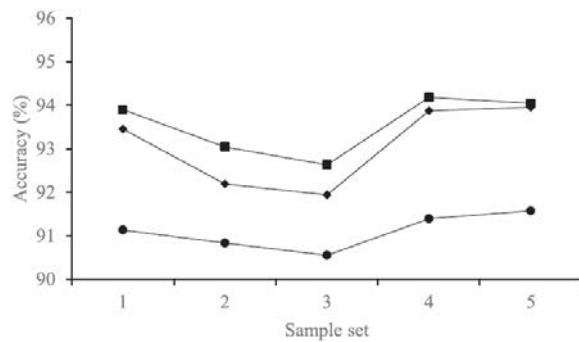
To evaluate the CNN, the research team collected various features from different layers to activate feature fusion which is then fed to the SVM classifier. In figures 4.30(α') and 4.30(β') we see the individual performance of each layer on both datasets mentioned before. From all, the lower sampling layer S4 is the one achieving the best performance in all aspects. Next follows the evaluation of the strong feature CNN-SVM model with batch size set to 30 and learning rate to 0.01. For training of SVM, C-SVC and radial kernel function were used. Looking at the results in the figures 4.31(α') and 4.31(β') that are achieved via 5-fold and 10-fold cross-validation accordingly, we understand that the CNN-SVM has better Acc than the individual models do. Standard deviation also reveals the same idea. Also, comparing the

Accuracies (%) and sensitivity (%) of 2-class classification using private dataset.

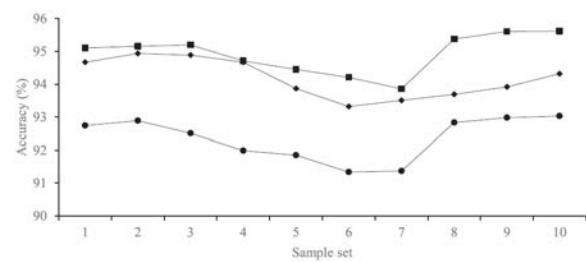
CNN layer	Accuracy	Sensitivity	Specificity
Convolution layer C1	90.4	90.0	90.2
Lower sampling layer S2	94.2	92.0	92.3
Convolution layer C3	93.2	91.8	91.7
Lower sampling layer S4	94.5	93.0	93.5

Accuracies (%) and sensitivity (%) of 2-class classification using Herlev.

CNN layer	Accuracy	Sensitivity	Specificity
Convolution layer C1	97.5	97.7	98.0
Lower sampling layer S2	98.5	97.3	98.3
Convolution layer C3	98.9	98.6	98.8
Lower sampling layer S4	99.3	98.9	99.4



(α') Accuracy of three models using 5-fold cross-validation from the values recorded in [30].



(β') Accuracy of three models using 10-fold cross-validation from the values recorded in [30].

two validation methods, 10-fold seems to have a slight advantage. To correlate the iteration times to the conditions of the model, they run the experiment twice, which means 20 training iterations and 500 iterations for evaluation. We see how the results are formed for 256 iteration in fig.4.32. We can see the impact that the SVM classifier has to the conventional CNNs. Since in the first iteration the weight vector of the CNN have random value, it makes sense that it starts from such low Acc. The best Acc the CNN-SVM reaches is 94.2% while the strong feature variant reaches 94.5%.

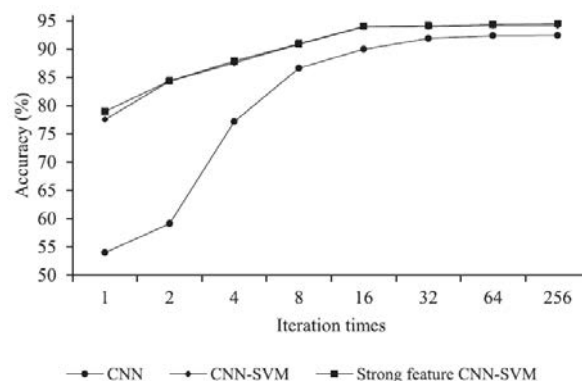


Figure 4.32: Relation between iteration times and model accuracy from CNN-SVM method [30].

On a comparison of activation functions, the research team invested time to test three popular variants: *Sigmoid*, *ReLU* and *Tanh*. Taking a look at figure , we see the slow response of the *Sigmoid* case which comes from the large computation process. *ReLU* shows a non-linear trajectory which means that there is the possibility of setting the negative value to 0, ensuring the sparsity of the model. Also, non-zero values are getting trained through gradient thus securing robustness. Thus, the research team switched to this activation function. The processing time in all three cases were:

- Sigmoid: 47s
- Tanh: 68s
- *ReLU*: 41s

4.7 DeepCervix

Moving one year forward, to 2021, we analyze the paper published from Rahaman et al.[31] with their DL CAD system about cervical cancer, named DeepCervix (fig.4.33). On the workflow diagram of the model, we see six distinct steps. These are the following:

1. The PAP smear images come from their database (SIPaKMeD, Herlev) as input.
2. Preprocessing follows where two stages of data augmentation tasks are implemented. The first one uses geometric manipulation, for example affine transformations, adding Gaussian or Laplace noise, canny filter, edge detection, colour filter, brightness and contrast adjustments to produce more samples for training. The SIPaKMeD data become 6 times more from these experiments. In the second one, the research team use the Keras "ImageDataGenerator" that involves augmentation, randomly reconstructing the images of the dataset during the training phase.
3. Next follow four DL models: VGG16, VGG19, XceptionNet, ResNet50. These are managing the feature extraction part of the architecture.
4. The feature fusion network (FFN) that comes next takes the features that come from the last layer of each DL model, creating the feature arrays of 1024 elements. These are then going into batch normalization from a dense layer which is followed from a dropout layer.

5. The images that are left for testing are then provided for classification.

6. Last comes the performance evaluation with Prec, Rec, F1 score and Acc.

This the first cervical cancer diagnosis model we see that uses HDFS techniques.

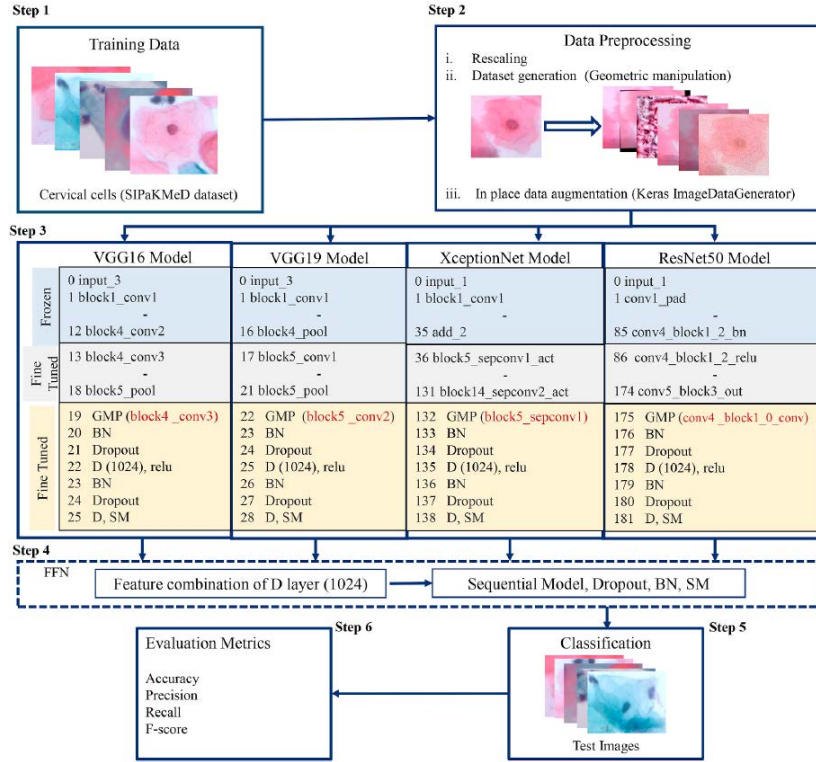


Figure 4.33: Workflow diagram of the proposed DeepCervix model [31] (global max pooling (GMP), batch normalization (BN), dense layer (D), Softmax (SM)).

The first handling technique used on the data to increase the number of samples is rescaling, where the developers take the SIPaKMeD BMP-formatted images that range from (71×59) to (490×474) and move them to (224×224) for all CNNs. Next follow a number of geometric transformations that add up to the previous processing.

- Affine Transformation (AT): Changes a pixels coordinates to a new position.
- Contrast Limited Adaptive Histogram Equalization (CLAHE): CLAHE is a method we have found before, which enhances the contrast of the areas produced from histogram equalization to reveal the bright and dark sectors.
- Edge Detection (ED): Taken from the "imgaug" API used, the "EdgeDetect" an "DirectEdgeDetect" functions do what their names suggest, providing edges detected from random angles.

- Canny filter (CF): The image are going through Sobel filter, making edges more visible.
- Photometric Transformation (PT): PTs mix the values of the colour channels such as the grayscale, hue and saturation.
- Contrast Adaptation (CA): Performs contrast and brightness changes.

An additional performance boost comes from Keras' "ImageDataGenerator" API which applies random transformation during training, producing a better training routine as the samples are unpredictable. For this to happen correctly, the feature-wise center has to be set to false and the rotation range to 5 degrees. Also, horizontal and vertical flips are activated, brightness can move from 50% to 130% and the channel can shift.

In terms of methods used, we see some familiar ones used here, as DL and transfer learning. One we haven't mentioned is late fusion, a technique that is based on weights from classification and how good that is. The weights come from the aforementioned DL models (VGG16, VGG19, ResNet50, XceptionNet) and the evaluation comes in the form shown below:

$$\sum_{m=1}^X E(m, n) = \max_{n=1}^Y \sum_{m=1}^X E(m, n)$$

where X is the number of classifiers and Y is the number of classes.

Since feature fusion (FF) has been trusted before in other methods. In serial FF method, two features are concatenated into a single feature ($F_s = (x + y)$). In parallel, FF makes a complex for every transformation ($F_p = F_1 + iF_2$). To prevent any overfitting problems, dropout and batch normalization layers are introduced.

In the SIPaKMeD dataset there are 4049 cervical samples of which, 60% used for training, 20% for validation and other 20% for testing. There are three different classification types: 5-class (superficial parabasal, koilocytotic, metaplastic, dyskeratotic), 3-class (normal, abnormal, benign), 2-class (normal, abnormal). The difference between them can be seen in fig.4.34.

Starting the accuracy comparison from the simplest, binary classification, the VGG16 model gets the best overall performance from the four DL models, reaching Acc of 99.85% with MVDC based LF getting comparable results. The least precise outcome is from the XceptionNet, with Acc of 98.31%.

Next follows the 3-class classification, where the VGG16 shows the most promising results again with Acc of 97.90% when the XceptionNet does not contribute as well as that, with Acc

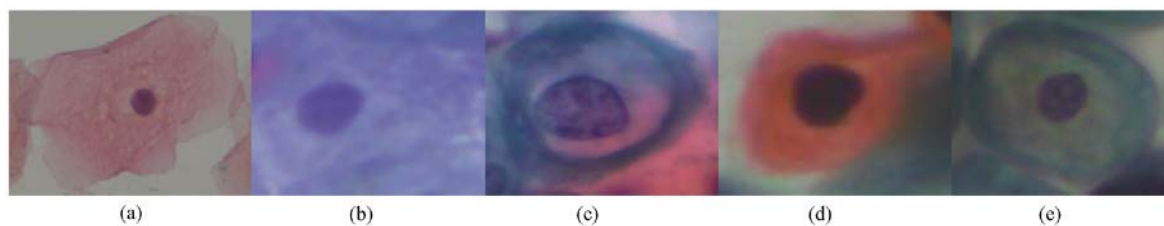


Figure 4.34: An example of SIPaKMeD database in five categories: (a) Superficial-Intermediate, (b) Parabasal, (c) Koilocytotic, (d) Dyskeratotic, (e) Metaplastic.

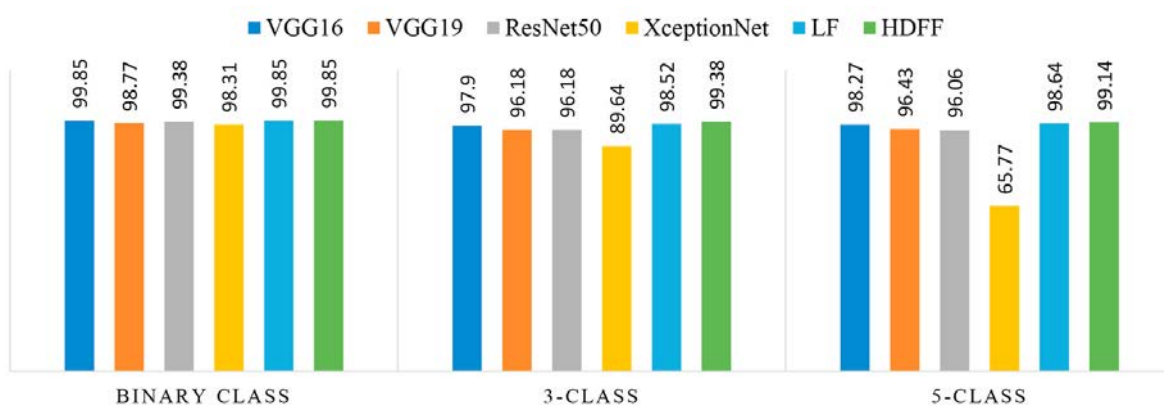


Figure 4.35: Performance comparison among different TL models with HDFS and LF methods. The figure is taken from [31].

of 89.64%. Additionally, LF obtains an Acc of 98.52% while the advanced HDFS proposed from the research team shows the best results, having 99.38% precise outcomes.

Moving to the most complex classification technique (5-class) we see the dominance of DeepCervix, which consistently performs accurately, as this time the 99.14% Acc that we get stays clear ahead from the rest, with second being the LF method and XceptionNet completely losing its performance, reaching 65.77% Acc. The visualization of the results that the research team obtained can also be seen in figure 4.35.

Moving to the SIPaKMeD dataset, we are now comparing the proposed method to a number of other models that exists and are developed around the current period of time (fig.4.36). Amongst them, DeepCervix again takes the cake both for binary and 5-class classification with 1.60% and 0.19% difference.

A significant difference between the individual mentioned DL models and the proposed one is that training time for each DL model takes around 6h for 100 epochs while HDFS takes only a few minutes with a rate of 3s per epoch.

Comparison of classification accuracies on the SIPaKMeD dataset.

Method	Class	Accuracy
CNN	5-Class	95.35%
Graph convolutional network	5-Class	98.37%
DenseNet-161	5-Class	98.96%
Bagging Ensemble	2-Class	98.25%
Classifier	5-Class	94.09%
HDFP	2-Class	99.85%
	3-Class	99.38%
	5-Class	99.15%

4.8 MFFOA-DL3

Modified Firefly Optimization Algorithm with DL enabled or MFFOA-DL3 is a novel cervical cancer classification model for classification of PAP smear images introduced by Vaiyapuri et al.[32] in 2022. Before analyzing the steps of the proposed method, an overview of the workflow can be seen in fig.4.37.

The dataset used from the research team is familiar to us, as they chose to go with the Herlev database, a fact that makes the comparison between the methods that are taking a look at easier, as the data don't affect the judgement of the best choice.

Before getting into the model, the data go through a BF-based noise removal filter. This is employed to resolve the Gaussian obscure problem, as it couples radiometric distinction and Euclidean separation. The pixel estimation performed for the image's noise is:

$$D(x) = g_{\chi}^{-1} \sum_{m \in R_n} A_s(D_{mn}) * B_r(U_{mn}) * I(x)$$

where g_{χ}^{-1} is the normalized array of output function in NF, A_s and B_r represent spatial distribution. D_{mn} denotes the range kernel function weight and U_{mn} is the Euclidean distance photometrically of the pixel value in noisy environments. Putting the math behind the theory produces the following:

$$g_{\chi} = \sum_{m \in R_n} A_s(D_{mn}) * B_r(U_{mn})$$

$$A_s(D_{mn}) = \exp\left(\frac{D_{mn}^2}{2} * \sigma_s^2\right)$$

$$B_r(U_{mn}) = \exp\left(\frac{U_{mn}^2}{2} * \sigma_r^2\right)$$

Next follows the image segmentation to locate the regions with affected cells. In this method, the KE algorithm is used with threshold t which has range of 1 to 255 considering the 8-bit

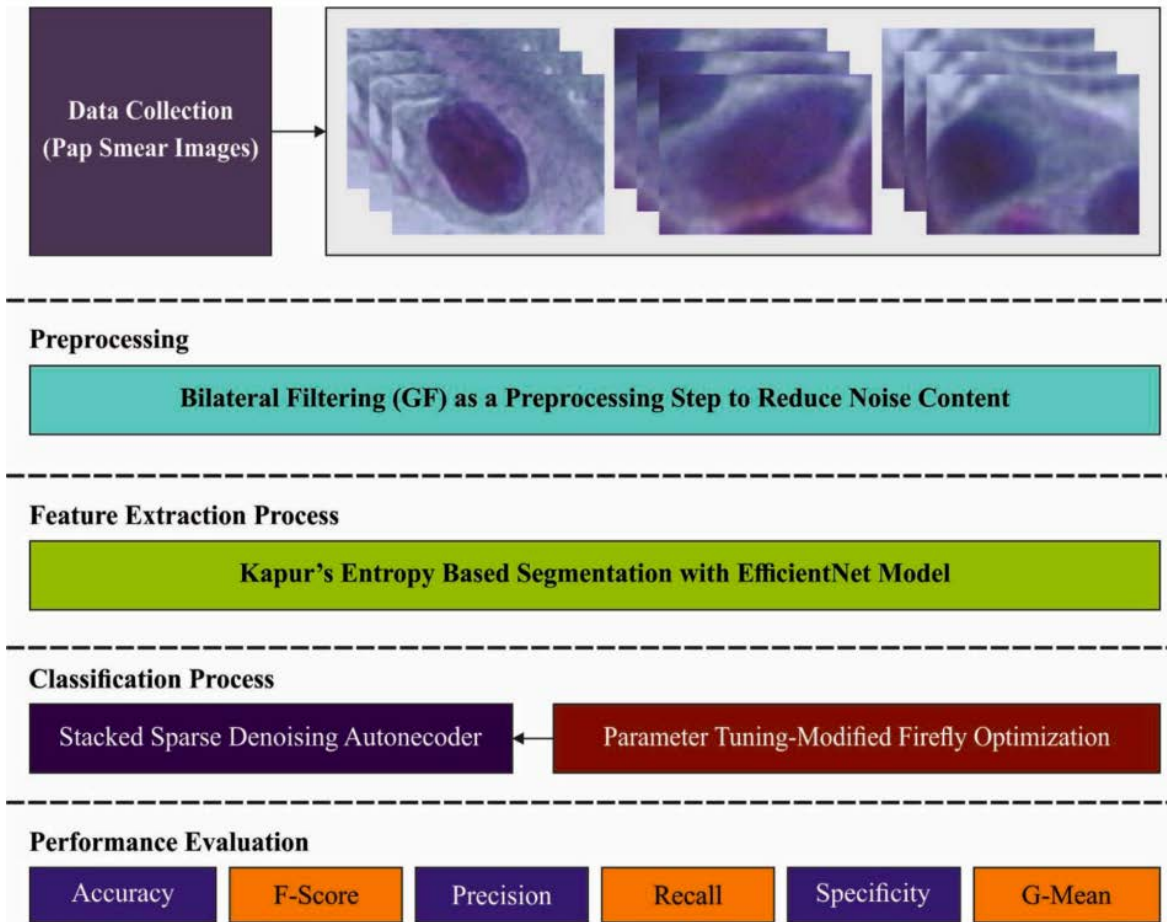


Figure 4.37: Overall block diagram of MFFOA-DL3 technique [32].

nature of the images. What it does is that it splits an image into E_0 and E_1 to maximize the subsequent process.

$$F(t) = E_0 + E_1$$

$$E_0 = -\sum_{i=0}^{t-1} \frac{X_i}{T_0} \times \ln\left(\frac{X_i}{T_0}\right), \text{ where } X_i = \frac{X_i}{T}, T_0 = \sum_{i=0}^{t-1} X_i$$

$$E_1 = -\sum_{i=0}^{L-1} \frac{X_i}{T_0} \times \ln\left(\frac{X_i}{T_0}\right), \text{ where } X_i = \frac{N_i}{T}, T_0 = \sum_{i=0}^{t-1} X_i$$

N_i depicts the number of pixels with gray value whereas T the total pixel count of the input image. Assuming that gray images have an intensity value between 0 and L-1, we find the n optimal thresholding value $[t_0, t_1, t_2, \dots, E_n]$ which subdivides the image to E_n parts.

$$F(t_0, t_1, \dots, t_n) = E_0 + E_1 + \dots + E_n$$

$$E_j = -\sum_{i=0}^{t_j-1} \frac{X_i}{T_j} \times \ln\left(\frac{X_i}{T_j}\right)$$

where $X_i = \frac{N_i}{T}$, $T_j = \sum_{i=0}^{t_j-1} X_i$, $j \in [0, n]$. After image segmentation, they introduced the EfficientNet model for feature extraction, which manipulates the resolution, width and depth

of the network for better performance. The image classification is done through the SSDA model [33]. The way the denoising autoencoder (DA) works is that (assuming x an image and y the resultant normalized image) the feed forward can be described as:

$$h(x) = s(Wx + b)$$

$$z(x) = s(W'h(x) + b')$$

If we consider the weight/bias vector as $\Theta = W, b, W', b'$ and $s(t)$ the sigmoid function, $h(x)$ the hidden layer, $z(x)$ the approximation of y . If, then, D is the training data than range from 1 to N , the sparse autoencoder (SDA) is trained with backpropagation with sparsity β as follows:

$$J(D; \Theta) = \frac{1}{N} \sum_{i=1}^N \frac{1}{2} \|z(x^i) - y^i\|_2^2 + \beta KL(\hat{\rho}||\rho) + \frac{\lambda}{2} (\|W\|_F^2 + \|W'\|_F^2)$$

$$KL(\hat{\rho}||\rho) = \sum_{j=1}^{\hat{\rho}} \rho \log_{\hat{\rho}_j} \rho + (1 - \rho) \log_{1-\hat{\rho}_j} (1 - \rho)$$

The target activation is represented by ρ and the average value of it as $\hat{\rho}$. Moving to SSDA, it is classified as a greedy layer-wise algorithm. Once every layer is pre-trained, the parameters of the L stacked SDA (2L hidden layer) are find-tuned through backpropagation simultaneously.

$$J_{SSDA}(D; \Theta) = \frac{1}{N} \sum_{l=1}^N \frac{1}{2} \|z(x^l) - y^l\|_2^2 + \frac{\lambda}{2} \sum_{l=1}^{2L} \|W_l\|_F^2$$

where W_l refers to the weight of the l -th layer in SSDA. From these, the developers invested in three stacked SDAs for their architecture. The also state the deployment of MFFOA [34]. The light intensity can be stated as follows:

$$Att_i = w_i e^{-\alpha q^2}$$

Then, we can say that the attraction of all the FFs is calculated to obtain the ideal parameters for both spatial and intensity kernels.

$$LightAtt = Actual_{light0} e^{-\alpha q^2}$$

where we state as actual light the original condition (attr = 0). The t -th element of spatial coordinate of m -th FF between two points m, n can be determined by their Cartesian distance.

$$q_{mn} = \sqrt{\sum_{\sigma=1}^s (M_m^t - N_n^t)^2}$$

Setting the next generation of FFs can be calculated, then, as:

$$M_m^{t+1} = M_n^t + \alpha(M_n^t - M_m^t) + \sigma(rand - \frac{1}{2})$$

σ denotes the irregularity parameter and $rand$ of a random number chosen per case.

$$rand = L_2(f_{max} - f_x) / (f_{max} - f_{avg}), f_x \geq f_{avg}$$

This optimization routine results to a fitness function.

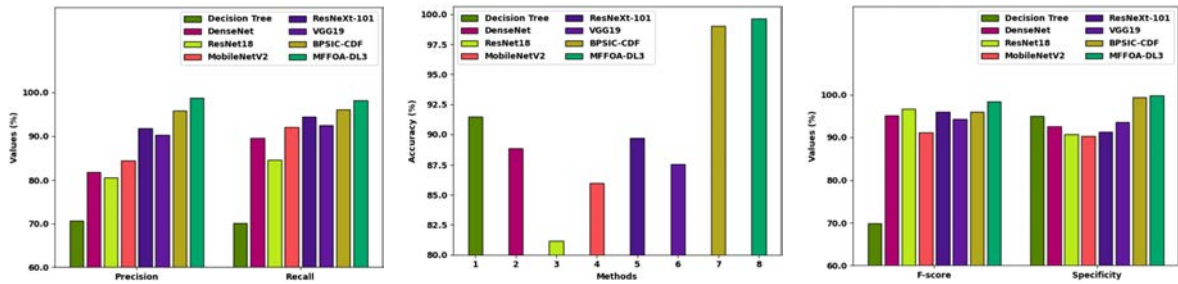
$$Fit(x_i) = ClassifierErrorRate(x_i) = \frac{\#MCImages}{\#Images} * 100$$

where *MCImages* stands for misclassified images.

The model was developed in Python 3.6.5 and evaluated on an Intel i5-8600k CPU, GeForce 1050Ti GPU and 16GB of RAM. The parameters set were: Learning rate (λ): 0.01, Drop-out parameter: 0.5, Batch size: 5, Number of epochs: 50, Activation function: ReLU.

The results obtained from MFFOA-DL3 method from the authors, with 200 epochs the model has performance of average $accu_y$, $prec_n$, $reca_l$, $spec_y$, F_{score} , and G_{mean} values such as 97.51%, 89.55%, 88.89%, 98.55%, 89.16%, and 93.50% respectively. Moving to twice the amount of epochs, the proposed MFFOA-DL3 technique obtained average $accu_y$, $prec_n$, $reca_l$, $spec_y$, F_{score} , and G_{mean} values such as 98.54%, 95.44%, 95.20%, 99.39%, 95.22%, and 97.24% correspondingly. Then on 600 epochs, the method accomplished average $accu_y$, $prec_n$, $reca_l$, $spec_y$, F_{score} , and G_{mean} values such as 99.56%, 98.29%, 97.92%, 99.75%, 98.09%, and 98.82% respectively. When reaching 800 epochs, the system attained average $accu_y$, $prec_n$, $reca_l$, $spec_y$, F_{score} , and G_{mean} values such as 99.63%, 98.60%, 97.98%, 99.78%, 98.25%, and 98.86% correspondingly. At the same time, with 1000 epochs, the proposed MFFOA-DL3 technique achieved average $accu_y$, $prec_n$, $reca_l$, $spec_y$, F_{score} , and G_{mean} values such as 99.66%, 98.80%, 98.13%, 99.80%, 98.43%, and 98.94% respectively. At last, with 1200 epochs, MFFOA-DL3 algorithm obtainable average $accu_y$, $prec_n$, $reca_l$, $spec_y$, F_{score} , and G_{mean} values such as 99.53%, 97.95%, 98.33%, 99.73%, 98.07%, and 99.02% correspondingly.

Lastly, we look how the MFFOA-DL3 model compares to existing models in terms of the Acc (fig.4.38(β')), Prec, Rec (fig.4.38(α')), F-score and specificity obtained (fig.4.38(γ')). On all axes, the MFFOA-DL3 has the best performance with second close coming the BPSIC-CDF method.



(α') $Prec_n$, $reca_l$ analyses results of MFFOA-DL3 technique and other recent algorithms. (β') $Accu_y$ analysis results of MFFOA-DL3 approach and other recent algorithms. (γ') F_{score} and $spec_y$ analysis results of MFFOA-DL3 approach and other existing algorithms.

4.9 Overall comparison

4.9.1 Accuracy comparison

If we compare all the above methods regarding the cervical cancer diagnosis (fig.4.39), we see that the research on this field has taken important steps forward from the first proposed method back in 2004. What we saw throughout the analysis is that there a number of different ways someone can go with feature extraction and classification of the image. Let's wait and see what the future holds for the progress of this field.

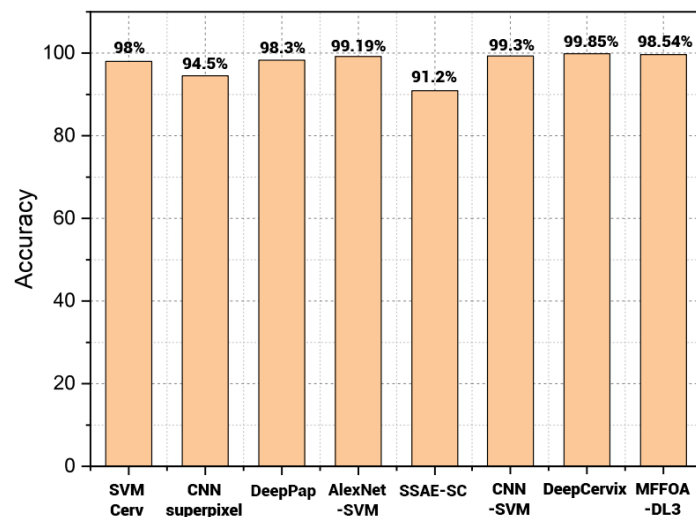


Figure 4.39: Comparative study of Acc for different Cervical Cancer detection and classification techniques.

4.9.2 Cost comparison

In order to make a more fair comparison between the models, we tested all of the above cervical cancer methods on the Herlev dataset we already mentioned which includes 917 samples. The testing was done on Matlab 2023a. The system includes an AMD Ryzen 5 5600X CPU, an NVidia RTX 3060 GPU and 16GB of RAM.

Taking a look on fig.4.41 that shows the number of hidden neurons per method, **DeepCervix** has the highest overhead by far with more than 24000 neurons more than the second worse (VGG16: 5096, VGG19: 9192, XceptionNet: 27664, ResNet50: 2268), which is very costly for the design of it but it also makes it the most accurate, as it is the most precise on the overall comparison (fig.4.40). Also, it has the lowest execution time, as seen in fig.4.42. Next follows **MFFOA-DL3** since it is a DL implementation with optimizations. **SSAE-SC** follows which includes a DNN (SSAE) which is then followed from a Softmax classifier for the two-step classification. **AlexNet-SVM** is the next method with most neurons with 6198 used which is the followed by **DeepPap** that has a higher delay than any of the other CNN implementations. **CNN-SVM** follows as the SVM doesn't add significantly the resources needed, followed by the **CNN superpixel** and the lowest complexity/highest delay comes from the basic **SVM Cerv** model designed in 2004.

When we test the methods with a more advanced and recent dataset, we see that the methods that are based on lower-complexity architecture have a tough time keeping up with the rest, as the 2004 method manages to reach only 85.7% Acc. Other than that, the DeepCervix method outperforms any other model, reaching an impressive 99.9% correct classification record.

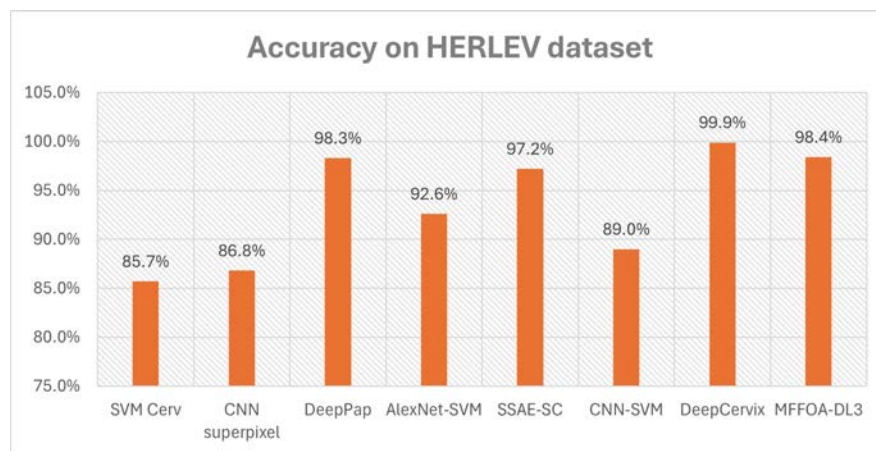


Figure 4.40: Comparative study of Acc for different Cervical Cancer detection and classification techniques on the Herlev dataset.

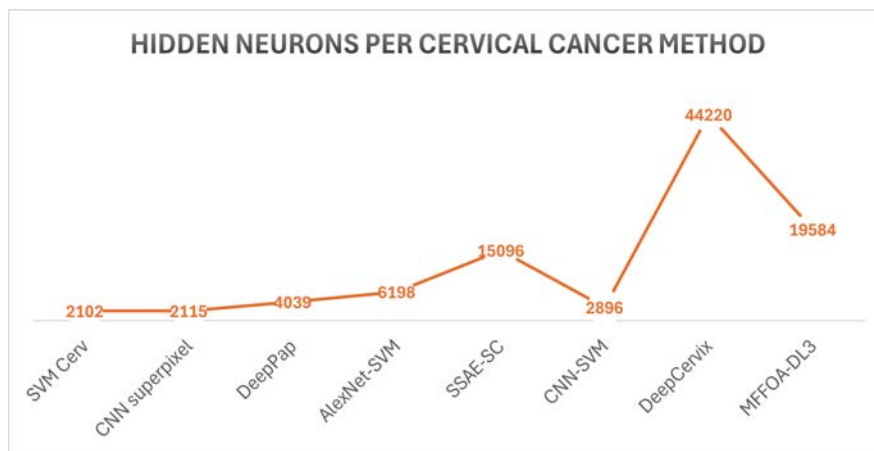


Figure 4.41: Comparative study of the number of hidden neurons for different Cervical Cancer detection and classification techniques on the Herlev dataset.

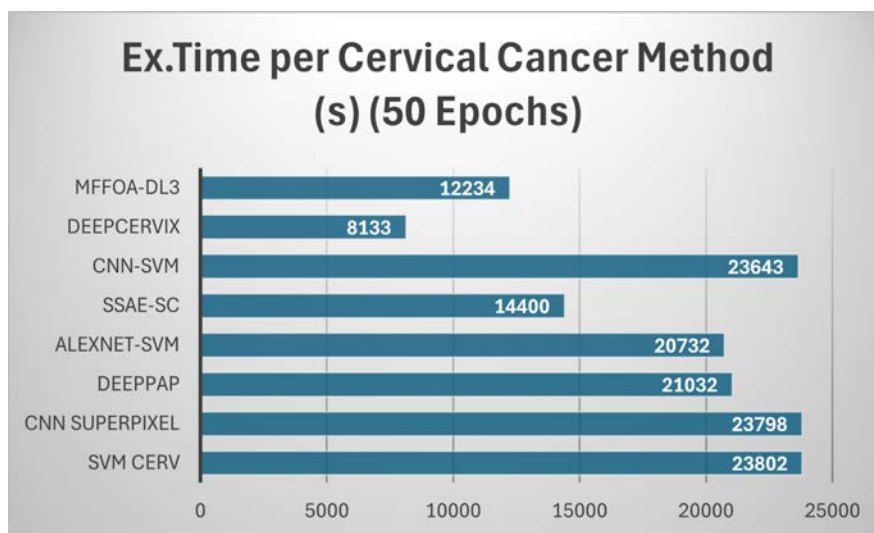


Figure 4.42: Comparative study of the execution time for different Cervical Cancer detection and classification techniques on the Herlev dataset.

Chapter 5

Conclusion

5.1 Overview and Conclusion

Summarizing on the topic of the thesis, as we saw, the uprise of ML and AI has taken world by storm, as it has emerged to aid various fields, making them more accessible for everyone and more efficient. Bioinformatics have also taken advantage of NN technology with various methods through the years, advancing early prediction of malicious cells on the body. In our thesis, we saw how imaging is used in AI technology on diagnosis of cervical cancer and brain tumor. Every method was measured by performance and cost, stating a number of algorithms, some of them (e.g. SVM) found in multiple and others showing a twist on how to approach the cause. The results showed that some methods seem nearly perfect, touching 100% accuracy and precision. Since we have come this far, the urgency of low-cost methods has become more unanimous, as it would help more clinics to have access to the latest proposed techniques. The latest methods that have been published are more focused on finding the balance between effectiveness and cost, as they gain a lot in terms of resources needed by sacrificing a relatively small amount of outcome precision. This thesis shows how the choice of algorithm, activation function and the general depth of a NN's layers affects how the model performs and we see that the trade-off between performance and architecture depth doesn't ruin the model.

5.2 Future work

In the future what we can see is how possible it is to further improve the already existing methods by using optimization techniques for inference models. A complete multiplier-less method, for example, would be a breakthrough in the medical community, as it would lower the architectural overhead significantly without taking a hit on the diagnosis of the patient, as plenty of multiplication happen during the process of an active inference models and multiplication happens to be the most cost-expensive operation. Another technique is quantization, which lowers the overhead by alternating the data type of weights and activation values to a smaller size. Also, pruning is often used to improve the architecture of NNs, as eliminates redundant or less important connections within a model. Even though all of the above optimizations would lower the resources need by a lot, they would also introduce the reduction of the method's performance severely. So, as we understand, it would be an interesting study to search for the balance of introducing such vast optimizations without making the method unusable.

Another aspect of future research could be the research of methods on cancer types that are not based on screening samples but on indications as blood tests, as stated in a paper published in 2020 from Chen et al.[35]. As we can see in fig.5.1, the performance of this model is pretty promising, no matter the different approach to the case study, as it reaches Spec of 95%.

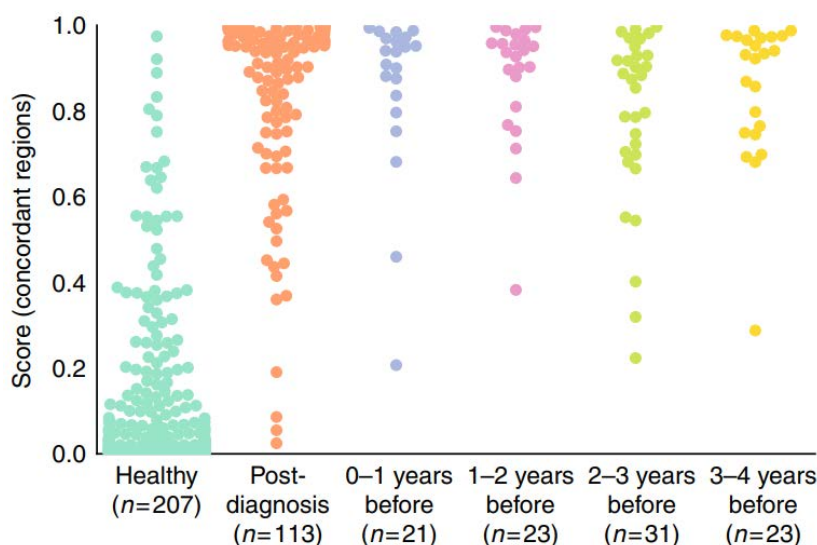


Figure 5.1: Logistic regression score recorded from the authors of the blood test-sampled method, found in [35].

Bibliography

- [1] KB Vaishnav and K Amshakala. An automated mri brain image segmentation and tumor detection using som-clustering and proximal support vector machine classifier. In *2015 IEEE international conference on engineering and technology (ICETECH)*, pages 1–6. IEEE, 2015.
- [2] Solmaz Abbasi and Farshad Tajeripour. Detection of brain tumor in 3d mri images using local binary patterns and histogram orientation gradient. *Neurocomputing*, 219:526–535, 2017.
- [3] Nicholas J Tustison, Brian B Avants, Philip A Cook, Yuanjie Zheng, Alexander Egan, Paul A Yushkevich, and James C Gee. N4itk: improved n3 bias correction. *IEEE transactions on medical imaging*, 29(6):1310–1320, 2010.
- [4] Terry S Yoo, Michael J Ackerman, William E Lorensen, Will Schroeder, Vikram Chakana, Stephen Aylward, Dimitris Metaxas, and Ross Whitaker. Engineering and algorithm design for an image processing api: a technical report on itk-the insight toolkit. In *Medicine Meets Virtual Reality 02/10*, pages 586–592. IOS press, 2002.
- [5] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.
- [6] Deepak Ranjan Nayak, Ratnakar Dash, and Banshidhar Majhi. Development of pathological brain detection system using jaya optimized improved extreme learning machine and orthogonal ripplelet-ii transform. *Multimedia Tools and Applications*, 77:22705–22733, 2018.
- [7] Sonal Deshwal, Pravesh Kumar, and Sandeep Mogha. An improved jaya algorithm (ijaya) for optimization. In Rajesh Kumar, Ajit Kumar Verma, Tarun K. Sharma,

- Om Prakash Verma, and Sanjay Sharma, editors, *Soft Computing: Theories and Applications*, pages 665–674, Singapore, 2023. Springer Nature Singapore.
- [8] Jian Wang, Siyuan Lu, Shui-Hua Wang, and Yu-Dong Zhang. A review on extreme learning machine. *Multimedia Tools and Applications*, 81(29):41611–41660, 2022.
- [9] Pradeep Kumar Mallick, Seuc Ho Ryu, Sandeep Kumar Satapathy, Shruti Mishra, Gia Nhu Nguyen, and Prayag Tiwari. Brain mri image classification for cancer detection using deep wavelet autoencoder-based deep neural network. *IEEE Access*, 7:46278–46287, 2019.
- [10] Mohammad Omid Khairandish, Meenakshi Sharma, Vishal Jain, Jyotir Moy Chatterjee, and NZ Jhanjhi. A hybrid cnn-svm threshold segmentation approach for tumor detection and classification of mri brain images. *Irbm*, 43(4):290–299, 2022.
- [11] Nabil Neggaz, Essam H. Houssein, and Kashif Hussain. An efficient henry gas solubility optimization for feature selection. *Expert Systems with Applications*, 152:113364, 2020.
- [12] Essam H. Houssein, Mosa E. Hosney, Diego Oliva, Waleed M. Mohamed, and M. Hassaballah. A novel hybrid harris hawks optimization and support vector machines for drug design and discovery. *Computers & Chemical Engineering*, 133:106656, 2020.
- [13] Abdelazim G Hussien, Essam H Houssein, and Aboul Ella Hassanien. A binary whale optimization algorithm with hyperbolic tangent fitness function for feature selection. In *2017 Eighth international conference on intelligent computing and information systems (ICICIS)*, pages 166–172. IEEE, 2017.
- [14] Raheleh Hashemzahi, Seyyed Javad Seyyed Mahdavi, Maryam Kheirabadi, and Seyed Reza Kamel. Detection of brain tumors from mri images base on deep learning using hybrid model cnn and nade. *biocybernetics and biomedical engineering*, 40(3):1225–1232, 2020.
- [15] Milica M Badža and Marko Č Barjaktarović. Classification of brain tumors from mri images using a convolutional neural network. *Applied Sciences*, 10(6):1999, 2020.
- [16] Jun Cheng. brain tumor dataset. 4 2017.

- [17] Abdu Gumaei, Mohammad Mehedi Hassan, Md Rafiul Hassan, Abdulhameed Alelaiwi, and Giancarlo Fortino. A hybrid feature extraction method with regularized extreme learning machine for brain tumor classification. *IEEE Access*, 7:36266–36273, 2019.
- [18] Hossam H Sultan, Nancy M Salem, and Walid Al-Atabany. Multi-classification of brain tumor images using deep neural network. *IEEE access*, 7:69215–69225, 2019.
- [19] Rajat Mehrotra, MA Ansari, Rajeev Agrawal, and RS Anand. A transfer learning approach for ai-based classification of brain tumors. *Machine Learning with Applications*, 2:100003, 2020.
- [20] Mantripragada Yaswanth Bhanu Murthy, Anne Koteswararao, and Melingi Sunil Babu. Adaptive fuzzy deformable fusion and optimized cnn with ensemble classification for automated brain tumor diagnosis. *Biomedical engineering letters*, 12(1):37–58, 2022.
- [21] C Suguna and SP Balamurugan. An extensive review on machine learning and deep learning based cervical cancer diagnosis and classification models. *Journal of Computational and Theoretical Nanoscience*, 17(12):5438–5446, 2020.
- [22] Jiayong Zhang and Yanxi Liu. Cervical cancer detection using svm based feature screening. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 873–880. Springer, 2004.
- [23] Youyi Song, Ling Zhang, Siping Chen, Dong Ni, Baopu Li, Yongjing Zhou, Baiying Lei, and Tianfu Wang. A deep learning based framework for accurate segmentation of cervical cytoplasm and nuclei. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 2903–2906. IEEE, 2014.
- [24] Ling Zhang, Hui Kong, Chien Ting Chin, Shaoxiong Liu, Xinmin Fan, Tianfu Wang, and Siping Chen. Automation-assisted cervical cancer screening in manual liquid-based cytology with hematoxylin and eosin staining. *Cytometry Part A*, 85(3):214–230, 2014.
- [25] Ling Zhang, Le Lu, Isabella Nogues, Ronald M Summers, Shaoxiong Liu, and Jianhua Yao. Deeppap: deep convolutional networks for cervical cell classification. *IEEE journal of biomedical and health informatics*, 21(6):1633–1643, 2017.
- [26] Ritu Nayar and David C Wilbur. *The Bethesda system for reporting cervical cytology: definitions, criteria, and explanatory notes*. Springer, 2015.

- [27] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678, 2014.
- [28] Bilal Taha, Jorge Dias, and Naoufel Werghi. Classification of cervical-cancer using pap-smear images: A convolutional neural network approach. In *Medical Image Understanding and Analysis: 21st Annual Conference, MIUA 2017, Edinburgh, UK, July 11–13, 2017, Proceedings 21*, pages 261–272. Springer, 2017.
- [29] D Selvathi and A Aarthi Poornila. Deep learning techniques for breast cancer detection using medical image analysis. *Biologically rationalized computing techniques for image processing applications*, pages 159–186, 2018.
- [30] A Dongyao Jia, B Zhengyi Li, and C Chuanwang Zhang. Detection of cervical cancer cells based on strong feature cnn-svm network. *Neurocomputing*, 411:112–127, 2020.
- [31] Md Mamunur Rahaman, Chen Li, Yudong Yao, Frank Kulwa, Xiangchen Wu, Xiaoyan Li, and Qian Wang. Deepcervix: A deep learning-based framework for the classification of cervical cells using hybrid deep feature fusion techniques. *Computers in Biology and Medicine*, 136:104649, 2021.
- [32] Thavavel Vaiyapuri, Haya Alaskar, Liyakathunisa Syed, Eman Aljohani, Ahmed Alkhayyat, K Shankar, and Sachin Kumar. Modified metaheuristics with stacked sparse denoising autoencoder model for cervical cancer classification. *Computers and Electrical Engineering*, 103:108292, 2022.
- [33] Yan Liu and Yi Zhang. Low-dose ct restoration via stacked sparse denoising autoencoders. *Neurocomputing*, 284:80–89, 2018.
- [34] Hao Qin, Ruifeng Yang, Chenxia Guo, and Weili Wang. Fault diagnosis of electric rudder system using psofoa-bp neural network. *Measurement*, 186:110058, 2021.
- [35] Xingdong Chen, Jeffrey Gole, Athurva Gore, Qiye He, Ming Lu, Jun Min, Ziyu Yuan, Xiaorong Yang, Yanfeng Jiang, Tiejun Zhang, et al. Non-invasive early detection of cancer four years before conventional diagnosis using a blood test. *Nature communications*, 11(1):1–10, 2020.