Medical Image Analysis Using Deep Learning
Brahim Ait Skourt

To cite this version:

HAL Id: tel-04108394
https://hal.science/tel-04108394
Submitted on 27 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Thèse pour l’obtention de DOCTORAT

Medical Image Analysis Using Deep Learning

Préparée par : Mr Brahim AIT SKOURT

Soutenue le 17 Janvier 2023 devant le jury :

<table>
<thead>
<tr>
<th>Nom et Prénom</th>
<th>Grade</th>
<th>Établissement</th>
<th>Qualité</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jamal KHARROUBI</td>
<td>PES</td>
<td>Faculté des Sciences et Techniques de Fès</td>
<td>Président</td>
</tr>
<tr>
<td>Abdelbaki EL BELGHITI</td>
<td>PES</td>
<td>Faculté des Sciences de Meknès</td>
<td>Rapporteur</td>
</tr>
<tr>
<td>ALAOUI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mohammed TALIBI</td>
<td>PES</td>
<td>Faculté des Sciences et Techniques de Fès</td>
<td>Rapporteur</td>
</tr>
<tr>
<td>ALAOUI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>El Mehdi ISMAILI</td>
<td>PH</td>
<td>Faculté des Sciences de Meknès</td>
<td>Rapporteur</td>
</tr>
<tr>
<td>ALAOUI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ilham CHAKER</td>
<td>PH</td>
<td>Faculté des Sciences et Techniques de Fès</td>
<td>Examineur</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nikola S. NIKOLOV</td>
<td>PES</td>
<td>Université Limerick -Irlande</td>
<td>Examinateur</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aicha MAJDA</td>
<td>PES</td>
<td>Faculté des Sciences Juridiques, Économiques et</td>
<td>Directeur de</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sociales de Meknès</td>
<td>thèse</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

N° d’ordre : 5/2023
In memory of my loving mother
Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text.

April 2023
Acknowledgements

Today I submitted my doctoral thesis report. It has been a long and sometimes daunting journey over the past few years. It would not have been possible without the support of many people. This dissertation would not have been possible without the inspiration and support of a number of wonderful people - I thank and appreciate them all for being part of this journey and making this dissertation possible. I am most grateful to my thesis supervisor, Professor Aicha MAJDA. Without her enthusiasm, encouragement, support, and constant optimism, this thesis could hardly have been completed. I thank Professor Nikola S. NIKOLOV, who was my supervisor during my stay in Ireland as a visiting PhD student at the University of Limerick, for his contribution. I am forever grateful to my colleagues in the Intelligent Systems and Applications Laboratory for their friendship and support, and for creating a cordial working environment. I would also like to thank the rest of the thesis committee: Pr. Jamal KHARROUBI, Pr. Abdelbaki EL BELGHITI EL ALAOUI, Pr. Mohammed TALIBI ALAOUI, Pr. El Mehdi ISMAILI ALAOUI, and Pr. Ilham CHAKER for their insightful comments and encouragement, and also for all the questions that encouraged me to further expand my research in the future. Finally, my deep and sincere gratitude to my family for their continuous and unparalleled love, help and support. I am forever indebted to all members of my extended family for giving me the opportunities and experiences that have made me who I am. They have selflessly encouraged me to explore new directions in life and seek my own destiny. This journey would not have been possible without them, and I dedicate this milestone to them.
Abstract

Deep learning is a machine learning subfield that emerged from artificial neural networks, and made a tremendous impact on various fields, such as image processing, speech processing, natural language processing and more. In image processing, Convolutional Neural Networks (CNNs) are the most popular deep learning architectures, given the nature of their feature extraction phase that is tailored for image data in particular. In this thesis, we perform medical image analysis using deep learning, but with the objective of investigating and discovering new deep learning architectures to solve medical image issues, such as lung nodule classification or brain tumor segmentation. There are different angles from where we can explore medical image analysis using deep learning: detection, segmentation or classification of body organs, diseases, tumors and more.

On the one hand, in medical image segmentation using deep learning through CNNs have shown great performance. For this reason, we investigated the use of a well-known deep learning architecture called U-Net, to conduct a lung CT image segmentation. Our aim in this experiment, is to show how well can the U-Net perform an image segmentation relying on a small amount of data. Furthermore, we proposed a new multi-scale architecture based on some of the finest feature extractors, besides using attention mechanism to emphasize the targeted object. We adopted this new architecture for brain tumor semantic segmentation, and showed satisfactory semantic segmentation compared to some deep learning architectures tailored for image segmentation such as U-Net, Attention U-Net and Fully Connected Network (FCN).

On the other hand, performing medical image classification adopting CNNs is a trend in computer vision, given their way of delivering outstanding performances. The feature extraction part of a CNN is the key to its popularity. For that matter, we proposed a comparative study that includes the feature extraction part of a CNN and other common feature extraction methods, such as Principal Component Analysis (PCA), 2D-Discrete Fourier Transform (2D-DFT) and Restricted Boltzmann Machine (RBM). With this study, we showed that CNNs achieve their best with convolutions and pooling layers. Subsequently, we dived deeper in the feature extraction part of the CNN, to investigate the option of how to make a CNN more accurate. Therefore, we proposed two different pooling methods under
two separate works. In the first method, we fully mixed max and average pooling in one layer of a CNN, and showed its superiority over conventional pooling methods accuracy-wise and compared to other mixed pooling strategies in terms of time performance. The second proposed pooling method is an amelioration of the first one. We showed that adding a dropout function to our mixed pooling strategy, boosted its performance and outperformed all the mixing pooling methods that was compared with.

The results of this thesis represent some important findings to better understand how deep learning models can be adjusted to medical image processing.

**Keywords:** Deep Learning, Medical Image Analysis, Convolutional Neural Networks, Medical Image Segmentation, Medical Image Classification.
Résumé

L’apprentissage profond est un sous-domaine de l’apprentissage automatique issu des réseaux neuronaux artificiels, qui a eu un impact considérable sur divers domaines, tels que le traitement des images, le traitement de la parole, le traitement du langage naturel, etc. En traitement d’images, les réseaux de neurones convolutionnels (CNN) sont les architectures d’apprentissage profond les plus populaires, étant donné la nature de leur phase d’extraction de caractéristiques qui est adaptée aux données d’images en particulier. Dans cette thèse, nous effectuons une analyse d’images médicales en utilisant l’apprentissage profond, mais avec l’objectif d’étudier et de découvrir de nouvelles architectures d’apprentissage profond pour résoudre des problèmes d’images médicales, tels que la classification de nodules pulmonaires ou la segmentation de tumeurs cérébrales. Il existe différents angles à partir desquels nous pouvons explorer l’analyse d’images médicales à l’aide de l’apprentissage profond, la détection, la segmentation ou la classification des organes du corps, des maladies, des tumeurs et plus encore.


D’autre part, la classification d’images médicales adoptant les CNN est une tendance dans la vision par ordinateur, étant donné leur façon de fournir des performances exception-
nelles. La partie extraction de caractéristiques d’un CNN est la clé de sa popularité. Pour cette raison, nous avons proposé une étude comparative qui inclut la partie d’extraction de caractéristiques d’un CNN et d’autres méthodes d’extraction de caractéristiques communes telles que l’analyse en composantes principales (PCA), la transformée de Fourier à Discrète 2D (2D-DFT) et la Machine de Boltzmann Restreinte (RBM). Avec cette étude, nous avons montré que les CNN atteignent leur meilleur niveau avec des convolutions et des couches de pooling. Par la suite, nous avons explorer plus profondément la partie d’extraction de caractéristiques du CNN, afin d’étudier la possibilité de rendre un CNN plus précis. Par conséquent, nous avons proposé deux méthodes de pooling différentes dans le cadre de deux travaux distincts. Dans la première méthode, nous avons entièrement mélangé le pooling max et moyen dans une couche d’un CNN et avons montré sa supériorité sur les méthodes de pooling conventionnelles en termes de précision et comparé à d’autres stratégies de pooling mixtes en termes de temps de performance. La deuxième méthode de pooling proposée est une amélioration de la première. Nous avons montré que l’ajout d’une fonction dropout à notre stratégie de pooling mixte, a augmenté ses performances et a surpassé toutes les méthodes de pooling mixte avec lesquelles elle a été comparée.

Les résultats de cette thèse représentent des découvertes importantes pour mieux comprendre comment les modèles d’apprentissage profond peuvent être adaptés au traitement des images médicales.

**Mots Clés:** Apprentissage Profond, Analyse d’Images Médicales, Réseaux de Neurons Convolutifs, Segmentation d’Images Médicales, Classification d’Images Médicales.
# Table of contents

List of figures xii
List of tables xiv
List of Abbreviations xv
General Introduction 1

## I State-of-The-Art 8

1 Deep Learning for Image Analysis 9
   1.1 Introduction 9
   1.2 Deep Learning Overview 10
      1.2.1 Supervised learning 12
      1.2.2 Unsupervised learning 12
      1.2.3 Semi-supervised learning 13
   1.3 Convolutional Neural Networks 14
      1.3.1 Introduction 14
      1.3.2 Basic components of CNN 15
         1.3.2.1 Convolutional layer 16
         1.3.2.2 Activation functions 19
         1.3.2.3 Pooling layer 23
         1.3.2.4 Batch normalization 25
         1.3.2.5 Dropout 27
         1.3.2.6 Fully connected layers 28
   1.4 Deep Learning Applications in Medical Image Analysis 29
      1.4.1 Detection 30
      1.4.2 Segmentation 32
2 Lung CT Image Segmentation Using Deep Neural Networks

3 Multi-Scale ConvLSTM Attention-based Brain Tumor Segmentation

Conclusion

III Medical Image Classification

4 Feature Extraction Methods for Lung-Nodule Detection: A Comparative Deep Learning Study
5 Contributions to Mixed-Pooling layers

5.1 Introduction ......................................................... 75
5.2 Related work ....................................................... 76
5.3 Fully Mixed Max-Average Pooling for Convolutional Neural Networks ....................................................... 80
  5.3.1 Fully mixed max-average pooling ........................................ 80
  5.3.2 Results and discussion .............................................. 81
5.4 Mixed Pooling-Dropout for Convolutional Neural Networks Regularization ....................................................... 84
  5.4.1 Mixed pooling dropout .............................................. 84
  5.4.2 Experimental results .................................................. 86
    5.4.2.1 Mixed pooling dropout vs Max, Average and Stochastic pooling ........................................ 87
    5.4.2.2 Mixed pooling dropout vs Max pooling dropout ........................................ 87
    5.4.2.3 Dropout rate ..................................................... 91
5.5 Conclusion .......................................................... 92

Conclusion .............................................................. 94

General Conclusion ..................................................... 96

References .............................................................. 100
List of figures

1.1 Basic MLP architecture. ............................................. 11
1.2 Visual cortex resemblance with convolutional neural network architecture. . 15
1.3 Basic Convolutional Neural Network architecture. .......................... 16
1.4 Simple convolution operation. ........................................ 17
1.5 $1 \times 1$ convolution. .............................................. 17
1.6 Flattened convolution, X, Y and Z are the width height and depth respectively 18
1.7 Cross-channel convolution. .......................................... 18
1.8 Depth-wise convolution. ............................................. 18
1.9 Grouped convolution. ................................................ 19
1.10 Shuffled-grouped convolution. ..................................... 19
1.11 Sigmoid activation function representation. ............................... 20
1.12 Hyperbolic tangent activation function representation. .................... 21
1.13 ReLU activation function representation. ................................ 21
1.14 Leaky ReLU activation function representation. ............................ 22
1.15 Swish activation function representation. ................................ 23
1.16 Neural network transition from basic to dropout. ............................ 28

2.1 The U-Net architecture. ............................................... 45
2.2 Rectified Linear Units (ReLU), presented in (2.1). .......................... 46
2.3 The way that U-net architecture works is that it takes the input image to generate the corresponding map for segmentation. ....................... 46
2.4 Experimentation Results ............................................... 47

3.1 Inception block. ...................................................... 53
3.2 SE-inception block. .................................................. 53
3.3 ResNet block. ....................................................... 53
3.4 SE-ResNet block. ................................................... 54
3.5 Spatial Attention Module. ............................................ 54
List of figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.6</td>
<td>Overview of our proposed architecture.</td>
<td>55</td>
</tr>
<tr>
<td>3.7</td>
<td>Segmentation results sample: (a) is the input MRI images, (b) is the ground truth and (c) is the segmentation results using our proposed architecture.</td>
<td>58</td>
</tr>
<tr>
<td>4.1</td>
<td>A sample from the lung nodule data set used in our experimentation.</td>
<td>70</td>
</tr>
<tr>
<td>4.2</td>
<td>Overview of the data processing flow (from data pre-processing to classifica-</td>
<td>71</td>
</tr>
<tr>
<td></td>
<td>tion).</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Accuracy charts for the four different feature extraction methods in validation and training.</td>
<td>74</td>
</tr>
<tr>
<td>5.1</td>
<td>Illustration of a pooling operation.</td>
<td>77</td>
</tr>
<tr>
<td>5.2</td>
<td>Fully mixed max-average pooling block.</td>
<td>80</td>
</tr>
<tr>
<td>5.3</td>
<td>The convolutional neural network architecture adopted for all the used models. Note that the pooling layer is replaced by the convenient pooling operation regarding each one of the six models.</td>
<td>81</td>
</tr>
<tr>
<td>5.4</td>
<td>A sample from the lung nodule data set used in our experiments.</td>
<td>82</td>
</tr>
<tr>
<td>5.5</td>
<td>The three architectures adopted for the experimentation. Mixed-pooling-</td>
<td>88</td>
</tr>
<tr>
<td></td>
<td>dropout first version (a), second version (b) and max-pooling-dropout (c).</td>
<td></td>
</tr>
<tr>
<td>5.6</td>
<td>Model accuracy and loss charts for our proposed dropout-pooling method</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td>(v1 and v2) and max-pooling-dropout.</td>
<td></td>
</tr>
<tr>
<td>5.7</td>
<td>Training accuracy by retaining probability for our proposed mixed-pooling-</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>dropout method (v1 and v2) and max-pooling-dropout.</td>
<td></td>
</tr>
</tbody>
</table>
List of tables

3.1 Proposed method’s DSC score compared to those of U-Net, Att-UNet and FCN. .............................................. 57
4.1 DNN accuracy for the four feature-extraction methods. .................. 72
4.2 DNN loss for CNN and 2D-DFT. .................................................. 73
5.1 Model accuracy and time performance for max, average and our proposed pooling method. ............................................. 83
5.2 Model accuracy and time performance for all mixed-pooling architectures including our proposed method. ....................... 83
5.3 Proposed method’s accuracy, sensitivity and specificity compared to those of Max, Average and stochastic pooling. ....................... 89
5.4 Models’ accuracy, loss and time performance in seconds for our proposed method (both versions) vs max-pooling-dropout. .......... 90
5.5 Mean-network vs. averaging strategy at test time for our proposed method. . 90
List of Abbreviations

AI . . . . . . . Artificial Intelligence
ANN . . . . . . Artificial Neural Network
API . . . . . . Application Programming Interface
BM . . . . . . Boltzmann Machine
CAD . . . . . . Computer Aided Diagnosis
CD . . . . . . Contrastive Divergence
CNN . . . . . . Convolutional Neural Network
ConvLSTM . . Convolutional Long Short Term Memory
CPU . . . . . . Central Processing Unit
CRF . . . . . . Conditional Random Field
cSE . . . . . . Channel Squeeze and Excitation
CT . . . . . . Computed Tomography
DBN . . . . . . Deep Belief Network
DFT . . . . . . Discrete Fourier Transform
DL . . . . . . Deep Learning
DNN . . . . . . Deep Neural Network
DSC . . . . . . Dice Score Coefficient
ED . . . . . . Edema
EM . . . . . . Expectation Maximization
ET . . . . . . Enhancing Tumor
FC . . . . . . Fully Connected
FCN . . . . . . Fully Convolutional Networks
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFT</td>
<td>Fast Fourier Transform</td>
</tr>
<tr>
<td>FMMAP</td>
<td>Fully Mixed-Max-Average Pooling</td>
</tr>
<tr>
<td>FPN</td>
<td>Feature Pyramid Network</td>
</tr>
<tr>
<td>GAN</td>
<td>Generative Adversarial Network</td>
</tr>
<tr>
<td>GP</td>
<td>Gated Pooling</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics Processing Unit</td>
</tr>
<tr>
<td>GRU</td>
<td>Gated Recurrent Units</td>
</tr>
<tr>
<td>HAANet</td>
<td>Hierarchical Aggregation Network</td>
</tr>
<tr>
<td>HGG</td>
<td>High Grade Glioma</td>
</tr>
<tr>
<td>HRNet</td>
<td>High Resolution Network</td>
</tr>
<tr>
<td>LGG</td>
<td>Low Grade Glioma</td>
</tr>
<tr>
<td>LIDC</td>
<td>Lung Image Database Consortium</td>
</tr>
<tr>
<td>LSTM</td>
<td>Long Short Term Memory</td>
</tr>
<tr>
<td>MaDP</td>
<td>Max Pooling Dropout</td>
</tr>
<tr>
<td>MCMC</td>
<td>Markov Chain Monte Carlo</td>
</tr>
<tr>
<td>MiDP</td>
<td>Mixed Pooling Dropout</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-Layer Perceptron</td>
</tr>
<tr>
<td>MMAP</td>
<td>Mixed Max Average Pooling</td>
</tr>
<tr>
<td>MP</td>
<td>Mixed Pooling</td>
</tr>
<tr>
<td>MRF</td>
<td>Markov Random Field</td>
</tr>
<tr>
<td>MRI</td>
<td>Magnetic Resonance Imaging</td>
</tr>
<tr>
<td>MSConvLSTM</td>
<td>Multi-Scale Convolutional Long Short Term Memory</td>
</tr>
<tr>
<td>NCR</td>
<td>Necrotic</td>
</tr>
<tr>
<td>NET</td>
<td>Non-Enhancing Tumor</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>RBM</td>
<td>Restricted Boltzmann Machine</td>
</tr>
<tr>
<td>ReLU</td>
<td>Rectified Linear Unit</td>
</tr>
<tr>
<td>RNN</td>
<td>Recurrent Neural Network</td>
</tr>
<tr>
<td>SDN</td>
<td>Stacked Deconvolutional Network</td>
</tr>
</tbody>
</table>
SGD . . . . . . Stochastic Gradient Descent
ResNet . . . . . . Residual Network
scSE . . . . . . Spatial-Channel Squeeze and Excitation
SENet . . . . . . Squeeze and Excitation Network
SE-inception . . Squeeze and Excitation inception
SE-ResNet . . Squeeze and Excitation Residual Network
SIFT . . . . . . Scale Invariant Feature Transform
SML . . . . . . Stochastic Maximum Likelihood
sSE . . . . . . Spatial Squeeze and Excitation
TC . . . . . . . Tumor Core
TPU . . . . . . Tensor Processing Unit
WT . . . . . . . Whole Tumor
General Introduction

Context

For the last few decades, *Artificial Intelligence* (AI) has become one of the main research fields in the world. Given its importance, AI is related to every aspect of our lives (health, education, food, sports . . . ). Humans have long dreamed of creating intelligent machines. Human intelligence only manifests itself through the power of acquiring all kind of information using natural human senses and processing it through the brain. In the seventies, simulating this kind of mechanism was considered science fiction, but nowadays AI can process huge amounts and different kinds of data in few seconds.

Today, AI is a thriving field with many practical applications and exciting current research topics. We are turning to intelligent solutions to automate routine processing, that is, to process information from different sources, mainly images and speech. In the early days, artificial intelligence quickly tackled and solved problems that are intellectually difficult for humans but relatively simple for computers, mainly problems that can be described by a list of formal and mathematical rules. As a result, the real challenge for artificial intelligence has been to solve problems that are difficult to describe formally, such as recognizing faces, recognizing spoken words or people from their voices. These are the types of problems that humans can solve intuitively.

There is a solution to these types of problems that allows computers to learn from experience and understand the world according to a hierarchy of concepts. This hierarchy, represents a collection of knowledge points that are connected in a deep way to form a graph with many layers. This concept is called Deep Learning (DL).

Today, AI is mainly manifested through its sub-fields, machine learning and deep learning. Deep learning was inspired by deep neurobiological structures of human speech and vision perception. Deep learning was introduced by Hinton and his team in 2006 [51], it is based
on improvements of Artificial Neural Networks (ANN) and the way they were trained. Deep learning made a revolution in many research fields such as medical domain. Therefore, medical image diagnosis is considered as a pattern recognition problem that could benefit from deep learning techniques. Nowadays, it has attracted great interest due to the high demand for Computer Aided Diagnosis (CAD) applications.

Because of the complex nature of human body organs when captured in images, medical image analysis is a complicated task. Even for human experts (radiologists) it is a challenging mission to process medical images with the naked eye. Thus, relying on advanced technologies to help in getting an accurate diagnosis is becoming strongly recommended. That is why we have CAD systems. A CAD system is an efficient tool designed to provide assistance for radiologists in interpreting medical data (mostly images) in order to deliver an accurate diagnosis for a disease. CAD systems are known for their reduced time to diagnosis, in addition to the most important aspects which is accuracy, i.e., CAD accurately diagnose a patient in a very reduced time compared to human beings.

Let’s take an example, for brain tumor diagnosis, a CAD system must go through the following steps:

− **Preprocessing**: Preparing data and making it standard for the machine, including separating the region of study from the useless area.
− **Detection**: Detection of the tumors that are present in the region of study.
− **Segmentation**: After tumors detection, they need to be segmented in order to separate them from other parts of the brain that might interfere in the decision making.
− **Classification**: Now that we have our patches containing brain tumors, we can perform a classification to decide if the tumors at hand are benign or malignant.

In order to get a precise diagnosis, the adopted algorithms to build this kind of CAD systems must be very robust and very fast. As we mentioned above, there are different types of algorithms that can serve the purpose of a CAD system, but how fast and how accurate?

It is known that medical images are very different from other types of images and need a special way of treatment. In addition, in the literature, the application of deep learning in computer vision problems usually encounters images coming from natural scenes. Thus, in this thesis, we are studying and investigating different deep learning techniques and their behaviour towards medical images. Moreover, we introduce new deep learning techniques
that may improve the way of processing medical image data.

Research questions

Our main goal is to examine and make enhancements to the performance of deep neural networks in processing medical image data sets. To reach this goal, we are dividing "medical image processing using deep learning" into: medical data segmentation and medical data classification relying on deep learning for both of them.

There are multiple steps to achieve a better classification accuracy or an efficient segmentation result: data preprocessing and preparation, feature extraction, then comes segmentation or classification. In this section, we identify the questions related to our study, the answers to which form the core of the work we present in this thesis:

Data preparation is one of the mandatory steps to achieve satisfactory results; can a conventional deep learning method perform an efficient segmentation by relying on good data preparation?

Feature extraction is one of the main steps to prepare data for any computer vision system, it consists of extracting only relevant characteristics from data for next phases. What are the feature extraction methods that can help our deep learning network achieve the best segmentation result?

CNNs are one of the state-of-the-art algorithms that made a huge impact in deep learning rise, it consists of a variety of feature extraction functions followed by a classifier. Medical image classification also relies on good feature extraction methods to reach higher accuracy, thus CNNs can be a good fit for this matter. Are convolution and pooling methods the best choice to optimal features extraction for CNNs? What are the enhancements we could make to CNN’s features extraction methods to improve its performance?

In this thesis, we focus more on the deep learning architectural side and not trying to solve medical problems. In other words, our main goal is to find new contributions in deep learning architectures that may be beneficial for computer vision problems in general. The choice of the medical domain is to show that if our findings can show great results while
processing medical images (that are a little bit complex compared to other types of images) we can for sure achieve promising results while dealing with other types of images.

**Research goals**

We provide in this thesis our findings under two computer vision problems:

**Medical image segmentation:**

- **Conventional deep learning methods:**
  
  In this chapter, we will conduct an experimentation involving a deep learning segmentation of the lung area from input CT scans of lungs. For that matter, we are going to investigate the performance of a conventional deep learning architecture called U-Net [118] using a publicly available data set LIDC-IDRI [4] containing lung CT scans.

- **Deep learning methods with advanced feature extraction methods:**
  
  For this work, we are going to perform a semantic segmentation of brain tumors using an advanced deep learning architecture. This architecture incorporates some state-of-the-art feature extraction methods that were behind the raise of CNNs, beginning from 2012 [68]. Furthermore, these methods are architecturally organized in a multi-scale way, besides, using Attention units to emphasize and put more focus on the tumor region.

**Medical image classification:**

- **Conventional deep learning methods:**
  
  Deep learning is feature extraction and classification combined in an end-to-end network. Therefore, their success proves that extracting good features leads to better accuracy. Given that CNNs have convolutions and pooling functions as feature extractors, we are comparing other feature extraction functions to those of CNN to show that convolution and pooling layers are the best fit for extracting relevant features in a CNN architecture.

- **Deep learning methods for classification with advanced feature extraction methods:**
  
  In [143], a paper called “going deeper with convolutions”, authors proposed a new convolutional block which showed great results in many computer-vision tasks. However, in this chapter of our thesis, we are going deeper with poolings, meaning that we are proposing new pooling strategies involving two of the state-of-the-art pooling functions max and
average, in order to boost the performance of CNNs in medical image processing in particular and in image processing in general.

**Contributions**

In this thesis we are presenting our contributions under two part:

- **Medical image segmentation:**

- **Medical image classification:**
  - "Brahim Ait Skourt, Nikola S. Nikolov, and Aicha Majda. "Feature-extraction methods for lung-nodule detection: A comparative deep learning study." 2019 International Conference on Intelligent Systems and Advanced Computing Sciences (ISACS). IEEE, 2019.". Proving that consecutive convolution and pooling function are the best fit for CNNs to reach higher accuracy results compared to PCA, 2D DFT and RBM.
  - "Brahim Ait Skourt, Abdelhamid El Hassani, and Aicha Majda. "Mixed-pooling-dropout for convolutional neural network regularization." Journal of King Saud University-Computer and Information Sciences 34.8 (2022): 4756-4762". Proposing a new mixed pooling method that relies on pooling and dropout which led to accurate results compared to conventional and other mixed pooling methods.
  - "Brahim Ait Skourt, Nikola S. Nikolov, and Aicha Majda. "Fully Mixed Max-Average Pooling for Convolutional Neural Network" accepted in international network of research and training inrt agadir 2022". Providing a study in which we compare the performance of conventional pooling methods with mixed-pooling methods and propose a new mixed-pooling strategy.
Thesis structure

In this thesis, we are covering many deep learning aspects performed in medical images in details. In fact, this thesis is presented under five parts with eight chapters organized as follows:

**Introduction:** This section presents the general introduction (current section).

**Part I:** In this, we are going to provide an overview of deep learning, and since our thesis is mainly about medical image processing, we will focus the presentation on the CNN architecture through its layers. Then we will briefly present the nature of medical images and the way deep learning can deal with them. Then, we will show through some state-of-the-art examples that with deep learning, processing medical images surely leads to great results in different types of image processing such as detection, segmentation and classification.

**Part II:** In this part, we are presenting two of our papers on the medical image segmentation filed. In the first chapter, we present our work where we perform lung CT image segmentation using a sophisticated deep learning architecture called U-Net. In the second chapter, we propose a novel architecture we called “Multi-Scale ConvLSTM Attention Neural Network” to perform brain tumor semantic segmentation. In this chapter we use some of the strongest feature extraction methods in the literature of deep learning in addition to Attention units in a multi-scale architecture.

**Part III:** this part is divided into three chapters. In the first chapter, we are providing a deep learning comparative study of some feature extraction methods such as 2D-DFT, PCA and RBM compared to the feature extraction layers in a standard CNN.

In the first section of the second chapter, we are presenting a pooling function comparative study and proposing a new one. Our proposed pooling method relies on mixing both max and average pooling methods in a specific way. Then, we compare its performance with conventional pooling functions such as max pooling and average pooling, then we compare its performance with other pooling strategies that involve mixing max and average pooling.

The second section of the second chapter is an extension of the latter section, where we are introducing a novel pooling method that mixes max and average pooling in one layer with the inclusion of dropout function in two different ways in order to enhance the performance of the network.
**Conclusion**: In this last section, we present a general conclusion of our thesis and future works.
Part I

State-of-The-Art
Chapter 1

Deep Learning for Image Analysis

Contents

1.1 Introduction ................................................. 9
1.2 Deep Learning Overview ................................. 10
  1.2.1 Supervised learning .............................. 12
  1.2.2 Unsupervised learning ...................... 12
  1.2.3 Semi-supervised learning ................... 13
1.3 Convolutional Neural Networks .......................... 14
  1.3.1 Introduction .................................. 14
  1.3.2 Basic components of CNN ................ 15
1.4 Deep Learning Applications in Medical Image Analysis .... 29
  1.4.1 Detection .................................. 30
  1.4.2 Segmentation .............................. 32
  1.4.3 Classification ................................ 33
1.5 Conclusion .............................................. 34

1.1 Introduction

Deep learning is one of the most active fields, not only in research domain but also in the industry. It is basically constituted from different types of deep neural networks such as CNNs, Deep Belief Networks DBNs, Recurrent Neural Networks RNNs… Their reputation came as a consequence of their ability to learn high levels of abstraction from significantly large
amounts of data, and yet accomplish great results compared to conventional machine learning algorithms. Deep learning can handle different types of data like images, audio, text and so on. In our case we deal with image data sets, thus we will only cover deep learning for image analysis generally and for medical image data sets specifically.

**CNN** is one of the deepest networks that showed very promising performance in the deep learning domain. This architecture is known for its ability to process image data sets in particular, very efficiently. In this part, we introduce an overview of deep learning domain, go through the **CNN** architecture and its components and present the impact of such algorithms in the medical image analysis domain.

### 1.2 Deep Learning Overview

Deep learning is a sub-field of machine learning that has raised rapidly over the last few years, it is represented by many architectures that have shown very promising results and have proven to be very effective in different application domains, due to their ability to deal with large amounts of data.

Deep neural networks are basically based on artificial neural networks, which try to simulate the human brain behavior by imitating the attitude of biological neurons. It started in 1943 when McCulloch proposed a linear threshold unit as a computational model with binary inputs and outputs [108]. Another type of ANN called perceptron was introduced in 1958 by Rosenblatt [119], which was considered a pioneering artificial neural network at that time. In 1969, Minsky et al. wrote a book called “Perceptrons” [96] that mainly revealed the limitations of the perceptron, Minsky showed in his book that it is impossible for the perceptron to learn a non-linearly separable function like XOR. Unfortunately, this book caused a significant disinterest and lack of funding for artificial neural networks research. Nevertheless, research on neural networks was revitalized with the introduction of the backpropagation algorithm [124] in the late 1980s which made them regain their strength.

A Multi-Layer-Perceptron, Fig.1.1, is an ANN with an input and output layer plus one or more hidden layers with multiple hidden units in each one of them.

To train an ANN, a backpropagation algorithm is performed with the Gradient Descent (GD) which is a first-order algorithm used for minimizing the error function applied to update the parameters, it has been successfully used for a long period to train ANNs, even with the long training time issue. The problem with GD is that it runs through all the samples in the training set to perform a single parameter update which is time consuming. Thus, the Stochastic Gradient Descent (SGD) was proposed to overcome this drawback, it consists of using a small sample of the training set to update the parameters instead of using the whole
systematic. Nowadays there are plenty of optimization algorithms for ANNs that have proven even more effective compared to SGD such as ADAM [67] and RMSprop [49]. Furthermore, in the optimization algorithms we specify a hyperparameter called the learning rate $\eta$, it controls the pace in which the parameters (weights) of the ANN are updated. It is an important component due to its role in speeding up the training process. Nonetheless, the value of the learning rate must be chosen wisely because a larger value may lead to divergence instead of convergence. On the other hand, a smaller value can get the training procedure to be stuck in a local minimum. Thus, ADAM and RMSprop have the ability to reduce the learning rate in an adaptive way, on the other hand, there are some commonly used solutions for reducing the learning rate during training according to a pre-defined schedule, such as Step-based decay, Time-based decay and Exponential decay which consist of periodically adjusting the learning rate.

ANNs have known many improvements since their start, but the most important one was in 2006 with the beginning of deep learning, when Hinton and his team proposed a new learning algorithm called the greedy-layer-wise [51] to train deep belief networks. Besides this outbreak, there are many reasons that helped deep learning algorithms to outstand other machine learning algorithms, one of which is that the depth of the architectures is no longer an issue given the invention of advanced technologies such GPUs and TPUs. On top of that, data has become more available than ever before. Machine learning, in particular Deep learning, approaches can be categorized to three subcategories: supervised, semi-supervised and unsupervised.
1.2.1 Supervised learning

It is a learning technique that uses labeled data to match a certain input with its corresponding output. The algorithm tries to minimize the error function by updating the network parameters until it gets to approximately generate the desired output. After the training, the algorithm should be able to get correct outputs for new data. In other words, the aim of supervised learning models is to accurately predict the correct output class for the newly unseen inputs.

Supervised learning algorithms usually try to solve two main problems: classification or regression. During the training phase, classification algorithms are given data points with an assigned label or class. The goal of classification algorithms is to take the input data and link it to its correct class, based on the training process learned beforehand.

Classification algorithms can be binary, i.e., two classes, or multi-class. For example, a classification of lung nodules as benign or malignant is considered binary, on the other hand a classification of different species of dogs is a multi-class classification.

There are many algorithms to solve classification problems. The choice of a specific algorithm for a specific problem depends on the data. Here are some popular classification algorithms ANNs, SVM, CNN, Random Forest... On the other hand, regression is considered as a predictive statistical process that attempts to find the correlation between dependent and independent variables. In opposition to classification, regression aims to predict continuous numbers such as time, sales, scores and so on.

There are several types of regression algorithms in the literature, from which we can mention: linear regression, logistic regression and polynomial regression.

1.2.2 Unsupervised learning

It is a learning strategy that tries to learn from data on its own without any supervision. It deals with unlabeled data to detect patterns in it and group them into clusters that share similar features. Unsupervised learning algorithms can be sub-categorized into two subcategories: Parametric and non-parametric unsupervised learning algorithms. The first subcategory assumes that sample data comes from a probabilistic distribution based on a set of fixed parameters. For example, the probability of any future observation can be easily generated when the mean and the standard deviation are known and the distribution is normal. The second subcategory does not require the modeler to make any assumptions regarding sample data which is assembled into clusters where each cluster is supposed to hold features of different categories that are present in the data.
1.2 Deep Learning Overview

Unsupervised learning algorithms are presented in the literature under clustering or association problems. Clustering is considered as assembling or organizing data entries into groups that are supposed to be similar in some way. The challenge in clustering is to find the best criteria in which data can be grouped. To do so, there are many proximity measures that help in finding a particular clustering solution such as Cosine distance, Jacard distance and Euclidean distance. Those proximity measures are adopted in various clustering algorithms, from which we can site K-means, Fuzzy K-means, Hierarchical clustering and Mixture of Gaussian.

On the other hand, association problems are different than clustering problems. By definition, clustering is grouping data objects in a way that objects in the same group are supposed to be similar than to those objects in other clusters. Whereas, association rules aim to find associations among objects within large commercial data sets. One popular example where association rules are applied is market basket analysis. The aim here is to associate grocery items that are frequently purchased.

1.2.3 Semi-supervised learning

As expected from its name, this type of learning falls between supervised and unsupervised learning strategies. Usually there is more unlabeled data than labeled data. In many cases, more unlabeled data can have a beneficial effect on the learning process like the case of Amazon Alexa as claimed by Jeff Bezos, the increased amount of untagged data helped in achieving more accuracy than before. One of the most popular learning techniques that adopts semi-supervised learning strategy is Reinforcement learning. This learning technique has gained a large popularity and has proven very robust and effective in the last few years. It is widely used in game-based algorithms and self-driving cars. Reinforcement learning is based on rewards and penalties given the action performed. Its aim is to maximize the overall reward. Reinforcement learning began in 2013 with google deep mind group [89] when they proposed an algorithm called deep Q-network that not only excels in playing the Breakout video game but outperforms all previous machine learning methods. Then the same group introduced AlphaGo [133] that outperformed the human level in the game of Go by defeating Lee Sedol in four out of five games. Thereafter, the same research group released a new version called AlphaGo Zero [134], it dramatically outperformed all the previous versions. Adopting reinforcement learning granted to AlphaGo Zero with just a little bit of knowledge about the game and few resources in terms of computation compared to previous versions to become the state of the art.
1.3 Convolutional Neural Networks

1.3.1 Introduction

Machine learning algorithms are able to understand the underlying correlation in the hidden data features, hence have the ability to make decisions without the need of any supervised instructions. Most of ML algorithms have been introduced to the literature since the beginning of the 1980s for the simulation of human behaviors in processing different types of data, such as speech and vision [111]. Nevertheless, ML algorithms have generally failed to achieve such an abstract level due to the way of handling data, therefore in late 1990s the challenging nature of computer vision tasks gave raise to a new type of Neural Networks called Convolutional Neural Networks that are suitable for understanding and processing image content. CNN has very important attributes that made it overcome the state-of-the-art in computer vision, those attributes are hierarchical learning, automatic feature extraction, multi-tasking, and weight sharing [149].

CNNs are one of the best algorithms for processing image data and have shown satisfactory performance in several image recognition tasks such as image segmentation, classification and detection [101], [63], [136]. Beyond the academic research field, CNNs have also captured attention in the industry companies such as Google, Facebook, Amazon and Microsoft, those companies have developed their own research groups and invested in exploring new architectures of CNN.

The strength of CNN resides in its ability to exploit spatial correlations in data. The architecture of CNN is composed of different learning stages; convolutional layers, non-linear activation units, subsampling layers and discriminative layers. CNNs are multilayered hierarchical networks, where each layer performs multiple transformations, for example the convolution operation helps in extracting useful features such as shapes and patterns from data points with different types of correlation. Then, its output is assigned to non-linear processing units to embed the non-linearity in the feature maps for the purpose of learning abstractions. The activation function produces different patterns of activations thus facilitates the learning process of semantic differences in input images. In some cases, the non-linear activation function is followed by normalization layer [59], to normalize the generated activations from the previous layer, i.e., maintain the mean activation close to 0 and the standard deviation close to 1. Then the subsampling is performed to summarize the results and prevent the input from geometrical distortions. The integrated automatic feature extraction makes CNNs have no need for a separate feature extractor. Therefore, CNN without thorough processing is able to learn high level representations from new presented images.
1.3 Convolutional Neural Networks

The very first idea of a CNN-like architecture was introduced under the name of Neocognitron [32], this architectural design was inspired by Hubel and Wiesel’s work in neuroscience, thus generally follows the basic structure of the human visual cortex in pattern recognition. For example, the way convolutional and subsampling layers extract features shows quite resemblance with V1 and V2 portion of visual cortex [28] as shown in Fig. 1.2. However, in 1990 Yann LeCun introduced an architecture inspired by Neocognitron for processing matrix-like topological data through his work intitled “Gradient-based Learning Applied to Document Recognition” [70], CNN first came to spotlight through this work.

![Visual cortex resemblance with convolutional neural network architecture.](image)

**Fig. 1.2.** Visual cortex resemblance with convolutional neural network architecture.

CNNs gained their popularity in image recognition due to their hierarchical feature extraction ability, which gives them the ability to capture different levels of features from low to high. Deep architectures mostly outstand shallow architectures when handling complex data, as a consequence of stacking multiple linear and non-linear processing units in a layer wise form. On the other hand, the availability of big data and advanced hardware have contributed significantly in the recent success of deep CNNs to achieve and outperform human level performance [61], [142].

### 1.3.2 Basic components of CNN

CNN is mostly considered as the widely used machine learning technique in several application domains, especially in vision related applications. CNNs have the advantage of possessing two major feature phases for processing grid-like data representations. For the feature extraction phase, CNNs can learn significant features. Then, the capability of feature classification. A typical CNN architecture is generally composed of convolutional layers with activation functions and pooling layers alternately, followed by one or more fully connected layers as shown in Fig. 1.3.
1.3 Convolutional Neural Networks

Fig. 1.3. Basic Convolutional Neural Network architecture.

Besides different mapping functions, normalization functions and dropout are also contained to enhance CNN performance [59], [68]. Next section discusses in details each component of CNN architecture and its role in achieving optimal performance.

1.3.2.1 Convolutional layer

A convolutional layer generates a map of abstractly convolved features, after performing convolution operation using slightly small parametrized filters of $N \times N$ shape. The nature of those filters defines the type of the resulted feature such as edges, horizontal lines, shapes and patterns. More formally, a convolution operation can be represented by the following formula:

$$y_k = f(W_k \ast x)$$  \hspace{1cm} (1.1)

where $x$ denotes the input image, $W_k$ is the convolution filter related to the $K^{th}$ feature map and the convolution operator is represented by the multiplication sign. The function $f()$ applied to the feature map is the activation function.

In general, the shape of input tensor remains invariant after performing a convolution transformation with same padding, leaving the reduction of feature maps size to the pooling layer. Nevertheless, it can be changed if a valid padding had been applied.

This operation can effectively simulate the task of visual cortex with the assistance of the non-linear transformation, which is represented by non-linear activation functions, which we will cover later. The purpose of performing non-linear transformation is to produce clearer contrast of meaningful features, thus granting the convolution operation to provide more abstract level in extracting features.

There are different types of convolutional layers considering their purpose in dealing with the input volume, from which we can site the following:
1.3 Convolutional Neural Networks

**Simple convolution**  It is the commonly used convolution type on a standard CNN, it is the dot product of the same filter with some width/height shape as shown in the following figure:

![Simple convolution operation](image)

**Fig. 1.4.** Simple convolution operation.

1 × 1 **convolution**  This specific type of convolution was first introduced in what is known by network-in-network architecture [83], then they were adopted in inception architecture [142] to serve the purpose of reducing the dimensionality in filter space, thus reducing the computation cost in the network. The following figure shows the way a 1 × 1 convolution reduces dimensionality in feature maps:

![1 × 1 convolution](image)

**Fig. 1.5.** 1 × 1 convolution.

**Flattened convolution**  It has the same reason of usage as the previous type but not only features dimension set to 1, also one of the other dimensions width or height. It consists of consecutive sequence of one-dimensional filters throughout all directions in 3D space to obtain promising results as conventional CNNs. Flattened convolution [62] has shown faster performance compared to standard convolution, due to the reduction of learning parameters.
1.3 Convolutional Neural Networks

Fig. 1.6. Flattened convolution, X, Y and Z are the width height and depth respectively

**Spatial and cross-channel convolution** It is widely adopted in Inception architecture, the main reason to use this kind of convolution is to split operations from cross-channel correlations into a series of independent operations, for example a $3 \times 3$ filter would be separately applied as $3 \times 1$ filter then followed by $1 \times 3$ filter.

Fig. 1.7. Cross-channel convolution.

**Depth-wise convolution** Unlike spatial separable convolutions, this kind of convolution works with filters that cannot be separated into two smaller filters. It consists of independently performing spatial convolution depth-wise, i.e. over each channel space. As shown in Fig.1.8:

Fig. 1.8. Depth-wise convolution.
1.3 Convolutional Neural Networks

**Grouped convolution**  It was firstly mentioned in AlexNet [68], the motivation behind using such convolutions is to optimize the network performance by reducing the computational complexity while dividing features into groups and computation over multiple GPUs. Next figure shows the working scheme of grouped convolution:

![Grouped convolution](image)

**Fig. 1.9.** Grouped convolution.

**Shuffled grouped convolution**  Coming from Shuffle-Net [170], the main idea is to avoid the side effect of only getting features from a certain channel that are only derived from a small subset of input channels. Fig.1.10 illustrates how a shuffled grouped convolution works:

![Shuffled grouped convolution](image)

**Fig. 1.10.** Shuffled-grouped convolution.

### 1.3.2.2 Activation functions

Activation functions represent a crucial component of deep convolutional neural networks, due to their role in determining the network’s output, its accuracy and its training computational efficiency, which plays a significant role in making or breaking a deep neural network. Activation functions have a huge impact on the CNN’s ability to converge. Nevertheless, the
choice of an activation function is a critical task given the fact that it might prevent CNNs from converging in the first place.

Activation functions are mathematical equations that define the nature of model’s output, they control the units that should be activated based on their relevance in making a model converge quickly. Moreover, activation functions help in normalizing each unit’s output to a range between 0 and 1 or between \(-1\) and \(1\). Due to the depth of recent CNNs, the computational strain has been increased on the activation functions, which led to the need for speed, hence the development of new functions such as Swish [110], ReLU [99] and its variants.

The use of non-linear activation functions has particularly become the mostly used in CNNs, which helps the network learn complex data features and provide accurate predictions.

There are three types of activation functions: Binary Step functions, Linear functions and Non-Linear functions. However, we are only going to cover the non-linear type, since it is the more sophisticated and the mostly used in CNNs.

**Sigmoid**  Sigmoid function [100] was widely used in neural networks in general, it non-linearly transforms the input values into a range between 0 and 1, as shown in the following figure.

![Fig. 1.11. Sigmoid activation function representation.](image)

The sigmoid function is characterized by its smooth gradient that prevents any jumps in the output values and it produces clear predictions given a slightly big input value which brings the output value to the edge of the curve. It is mathematically represented by the following formula:

\[
sig(x) = \frac{1}{1 + e^{-x}}
\]  \hspace{1cm} (1.2)

However, the sigmoid function has many drawbacks, such as vanishing gradient, output not zero centered and computationally expensive.
1.3 Convolutional Neural Networks

**Softmax**  The Softmax function is a generalization of sigmoid. Usually, the sigmoid function is used in a binary classification, whereas the Softmax function is used in a multiclass classification. The Softmax function can be represented as follows:

\[
\text{softmax}(x) = \frac{e^{x_i}}{\sum_{k=1}^{K} e^{x_k}}
\]  

(1.3)

Where \(i = 1, \ldots, K\).

**Hyperbolic tangent**  Usually referred to as Tanh, it is another non-linear function that has proven effectively compared to Sigmoid.

![Hyperbolic tangent activation function](image1)

**Fig. 1.12.** Hyperbolic tangent activation function representation.

The range of values is between \(-1\) and \(1\). Apart from that, \(tanh\) is very similar to \(sigmoid\), it is continuous and differentiable at all points. The \(tanh\) formula is represented as follows:

\[
tanh(x) = \frac{2}{1 + e^{-2x}} - 1
\]  

(1.4)

**ReLU**  It is another non-linear activation function that has proven very effective with the rise of deep neural network in the last decade.

![ReLU activation function](image2)

**Fig. 1.13.** ReLU activation function representation.
ReLU [99] gained popularity due to its speed and ability to not activating all the neurons at the same time, which provides computation efficiency when compared to Sigmoid and Tanh. ReLU is mathematically formulated as follows:

\[
ReLU(x) = \max(0, x)
\]

(1.5)

Nevertheless, ReLU has a disadvantage called the dying ReLU problem. When the inputs are negative or close to zero, the gradient of the function becomes zero which prevents the network from performing backpropagation to learn from data.

**Leaky ReLU** The leaky ReLU [158] aims to improve the ReLU function by presenting a solution to the dying ReLU problem.

![Leaky ReLU activation function representation.](image)

Instead of resulting a 0 for negative input values of \(x\), the function is defined as an extremely small linear component of \(x\) as expressed by the following formula:

\[
LReLU(x) = \begin{cases} 
0.01x, & x < 0 \\
x, & x \geq 0 
\end{cases}
\]

(1.6)

**Parametric ReLU** It is another variant of ReLU function [158] that aims to address the problem of gradient becoming zero when the input value is negative. It is the same as leaky ReLU, but instead of using a constant 0.01 we use a parameter \(a\) as shown in the following formula:

\[
PReLU(x) = \begin{cases} 
a x, & x < 0 \\
x, & x \geq 0 
\end{cases}
\]

(1.7)

**Swish** Swish is a newly discovered activation function by google research group. According to [110], it performs better than ReLU on deeper models with the same efficient computation.
1.3 Convolutional Neural Networks

The range of values is between negative infinity to positive infinity. The function is defined as:

\[
swish(x) = \frac{x}{1 + e^x}
\]  

(1.8)

1.3.2.3 Pooling layer

Pooling layers have been from the very beginning of CNNs, one of the building blocks and appear to play an important role in making CNNs very successful.

The integration of pooling layers into a CNN architecture is mandatory for many different reasons. First of all, pooling layers manage to reduce the dimensionality of input data, which have a huge impact on the computational cost of the model, it helps in dramatically decreasing it. Secondly, using pooling layers contribute in dealing with feature location sensitivity in input data. Besides that, pooling layers are involved in preventing overfitting. Their contribution to preventing overfitting resides in generating a lower version of features from input feature maps that preserves most of the important information while eliminating irrelevant features. Pooling layers are usually applied after performing activation non-linearity to convolution feature maps. There are many different variations of pooling layers, such as max-pooling, average pooling and some different versions of mixed pooling like Mixed-Pooling, mixed Max-Average-Pooling and Gated-Pooling functions. As to the existence of numerous pooling operations, the choice of a specific function for a specific problem had always been related to empirical studies as shown in [162]. Nevertheless, in [10] Boureau et al. presented some theoretical work which provides guidance as to which type of pooling operation can be adopted under specific circumstances.

Bellow, we present some of the popular pooling functions.
Max-pooling  It is the mostly used functions in the literature due to its ability in reaching better performance compared to standard functions. Max-pooling allows to select the maximum value among the values in the pooled region by performing the following formula:

\[
y_{ij}(x) = \max_{(p,q)\in R_{ij}} x_{k_{ij}}
\]  

(1.9)

Where \(x_{k_{ij}}\) represents the element at location \((p,q)\) covered by the pooling region \(R_{ij}\). Max-pooling is usually performed after the first or second convolution layer to guarantee the decrease of dimensionality.

Average-pooling  It is the second widely used pooling function. This operation is performed by taking the arithmetic mean of the values in the pooled area. It can be mathematically represented by the following formula:

\[
y_{ij}(x) = \frac{1}{|R_{ij}|} \sum_{(p,q)\in R_{ij}} x_{k_{ij}}
\]  

(1.10)

Where \(x_{k_{ij}}\) represents the element at location \((p,q)\) covered by the pooling region \(R_{ij}\).

Mixed-pooling  The choice between max and average pooling has always been difficult due to the closeness in terms of performance. Nevertheless, in many cases one is obviously better than the other. For that matter, Dingjun Yu et al. introduced a new pooling operation, Mixed-Pooling [162], that relies on randomly picking max or average pooling to be performed in a model. Mixed-pooling has shown satisfactory results, it outperformed both max and average pooling according to their experiments. This operation can be formally represented by the following formula:

\[
y_{ij}(x) = \lambda \max_{(p,q)\in R_{ij}} x_{k_{ij}} + (1 - \lambda) \frac{1}{|R_{ij}|} \sum_{(p,q)\in R_{ij}} x_{k_{ij}}
\]  

(1.11)

Where \(x_{k_{ij}}\) represents the element at location \((p,q)\) covered by the pooling region \(R_{ij}\) and \(\lambda\) can take 0 or 1, with 0 means average pooling operation is performed and 1 means max pooling in performed. In this way, the pooling regulation scheme becomes a probabilistic matter which helps in achieving better performance.

Mixed max-average-pooling  This method was proposed by Chen-Yu Lee et al. in [147]. It consists of proportionally mixing the two pooling operations, max and average, instead of just choosing one operation to perform. The mixed max-average pooling operation can be represented by the following formula:
1.3 Convolutional Neural Networks

\[
y_{ij}(x) = \alpha \max_{(p, q) \in R_{ij}} x_{kij} + (1 - \alpha) \frac{1}{|R_{ij}|} \sum_{(p, q) \in R_{ij}} x_{kij} \tag{1.12}
\]

Where \(x_{kij}\) represents the element at location \((p, q)\) covered by the pooling region \(R_{ij}\) and \(\alpha\) is a scalar representing the mixing proportion which specifies the exact amount of combination of max and average pooling, \(\alpha \in [0 - 1]\). We can see that mixed-pooling is a generalization of the mixed max-average pooling method when \(\alpha = 0\) or \(\alpha = 1\).

Gated-pooling  This method was proposed in the same paper as the last method as a solution to its drawback. Which is the non-responsiveness, i.e., the mixing proportion is fixed no matter what are the characteristics that constitute data. As a matter of fact, Gated-pooling was introduced to address the non-responsive behavior. Rather than fixing the mixing proportion, a gating-mask is learned that has the same spatial dimensions of the pooled region, then the inner product of this gating-mask and the region being pooled produces a scalar which is fed through a sigmoid function to generate the mixing proportion. This operation is translated using the following formula:

\[
y_{ij}(x) = \sigma(\omega^T R_{ij}) \max_{(p, q) \in R_{ij}} x_{kij} + (1 - \sigma(\omega^T R_{ij})) \frac{1}{|R_{ij}|} \sum_{(p, q) \in R_{ij}} x_{kij} \tag{1.13}
\]

Where \(x_{kij}\) represents the element at location \((p, q)\) covered by the pooling region \(R_{ij}\), \(\omega\) denotes values of the gated mask and \(\sigma\) is the sigmoid function represented by Eq.5.8.

1.3.2.4 Batch normalization

Deep Neural Networks are among the most if not the mostly sophisticated learning algorithms in machine learning field. Yet, training them remains a very challenging task, as they are sensitive to initial parameters configuration. For the purpose of overtaking this issue and to lead the model’s performance into accurate results, many optimization algorithms were adopted. Stochastic Gradient Descent (SGD) is one of these algorithms that has proven to be an excellent way of training DNNs. SGD has shown even more effective when used in mini-batches. However, as robust as SGD is, it still requires careful initialization of parameter values for the network. This seems to be an issue for the training process due to the correlation between the inputs to each layer and the parameters of all preceding layers. Therefore, as the network grows deep-wise, small changes to the network parameters make the layers’ inputs dramatically expand.

As a solution to address the beforementioned problem, one can immediately think of changing the distributions of layers’ inputs. However, as sophisticated as it seems, this
solution generates another problem called “Covariate Shift” [132], which can be handled via domain adaptation. Nevertheless, the Covariate shift problem appears to affect sub-parts of the model such as sub-networks included in the whole network or layers. When trying to adopt the same solution performed to the whole model, many dimensions of layers’ inputs will likely increase significantly which will slow down the convergence. In [59], the problem of changing distributions of internal units of a DNN is referred to as “Internal Covariate Shift”. Performing Batch Normalization in the training process offers a promise of eliminating it and guarantees a faster training. Batch Normalization is an approach that leads toward reducing the internal covariate shift by normalizing, the networks’ inputs, i.e. linearly transformed to have zero mean and unit variances and decorrelated. For optimization purposes, batch normalization is performed over mini-batches since normalizing each layer’s inputs is not optimal. Therefore, two simplifications are proposed regarding batch normalization in order to optimize the normalization process. Firstly, instead of jointly normalizing the features in layer’s inputs and outputs, each scalar feature is independently normalized as shown in Eq.1.14. For a layer of dimension $d$ and an input $x = (x^i, \ldots, x^k)$ each dimension is normalized as follows:

$$Z^i = \frac{x^i \text{Mean}(x^i)}{\sqrt{\text{Var}(x^i)}} \quad (1.14)$$

Where $\text{Mean}$ and $\text{Var}$ are computed over the training set. In practice, limiting the activation of each layer to a strict mean of 0 and variance of 1 can limit the expressiveness of the network. Hence, practically, batch normalization allows the network to learn $\gamma$ and $\beta$ parameters, which can convert the mean and variance into any desired value for the network. In other words, it allows to shift and scale the normalized value:

$$y^i = \gamma^i z^i + \beta^i \quad (1.15)$$

The used parameters in Eq.1.15 are learned along with the original model parameters. When setting $\gamma^i = \sqrt{\text{Var}(x^i)}$ and $\beta^i = \text{Mean}(x^i)$ can lead to restore the original activations if needed.

The second simplification resides on producing estimates of the mean and variance of each activation by each mini-batch since mini-batches are used in stochastic gradient training. Therefore, the used parameters for normalization are fully involved in the gradient back-propagation.

Batch normalization can be applied to any set of activations on the network. For the case of CNNs, the focus is on the transformations that consist of a related transformation followed by an element-wise non-linearity:
1.3 Convolutional Neural Networks

\[ Z = f(Wu + b) \]  

where \( W \) and \( b \) are the parameters learned from the model, and \( f(\cdot) \) is the nonlinearity function. This transformation includes both fully connected and convolution layers. The BN transformation is added just before the nonlinearity, i.e., normalize \( x = Wu + b \). The layer’s inputs could also have been normalized, but since \( u \) is probably the result of another nonlinearity, the shape of its distribution is likely to change during training, and limiting the first and second moments would not eliminate the covariate shift. On the contrary, \( Wu + b \) has a rather symmetrical and non-dispersive distribution, i.e., "Gaussian"; normalization should lead to activations with a stable distribution. Note that the bias \( b \) can be ignored, since its effect is cancelled by the subsequent mean subtraction. Therefore \( Z = f(Wu + b) \) is simply replaced by \( Z = f(BN(Wu)) \) where the BN is applied independently to every dimension of \( x = Wu \), with a distinct pair of learned parameters \( \gamma_k, \beta_k \) for each dimension.

In the case of convolution layers, the normalization needs to follow the convolution property, so that different elements of the same feature map are normalized in the same way at different locations. To achieve this, all activations are normalized in a mini-batch, in all locations. In this case, to perform BN, Values of \( x \) in a mini-batch \( B = x_1 \ldots m \), \( B \) is going to be a set of all the values of a feature map on both the elements of a mini-batch and the spatial locations. Thus, for a mini-batch of size \( s \) and feature maps of size \( i \times j \), the effective size of the mini-batch \( s' = s \cdot i \cdot j \) is used. This leads to learn a set of parameters \( \gamma_k \) and \( \beta_k \) per feature map, instead of per activation.

Batch Normalization has been applied to various state of the art models from the ImageNet classification such as inception [143], and has proven very robust and effective by overtaking the state-of-the-art accuracy as shown in [59].

1.3.2.5 Dropout

From the beginning, Neural Networks have faced several problems that made training very difficult, one of them is overfitting. There have been many attempts to address that issue, on the one hand the use of pooling layers and hyperparameters finetuning showed promising results in reducing the overfitting of such networks. On the other hand, regularization methods such as L1 and L2 [108] helped in reducing overfitting by keeping network’s weight as small as possible. Furthermore, N. Srivastava et. al. proposed a method called Dropout [119], which is a simple way of preventing neural networks from overfitting simply by introducing a new way of model combination.
For many machine learning methods, model combination seems to improve their performance. However, the computation increases with the increase of combined models, given the number of hyperparameters to fine-tune for each model. This task requires an important deal of computation. Additionally, model combination necessitates large amounts of data which may cause a problem since there may not be enough data to train different networks. Beside computation complexity and lack of enough data to train different models, even if it was practical to train several large networks, there is no way it can be optimal for inference given the importance of speed in test time. To resolve these issues, the dropout was proposed.

The idea behind dropout function is to provide an efficient way of approximately combining many different neural network architectures, hence addressing the overfitting problem too.

The dropout function works by randomly dropping out some units in the training process of neural networks. Dropping out a unit means temporally disabling its incoming and outgoing connections in the neural network. The dropout function introduced a new hyperparameter called “retaining probability” usually referred to as $p$, which is the probability of each unit to be retained. As stated in the original paper, the choice of $p$’s value can be done using a validation set or by simply setting to be 0.5. Although, the choice of an optimal value for $p$ appears to be a very important task. In [119], it is highly recommended for $p$ to be closer to 1 than to 0.5 for the input layers. While for hidden layers the optimal value is usually between 0.5 and 0.8 with $p = 0.5$ is most likely to be the optimal value. Nevertheless, the nature of data also plays a significant role in deciding the retaining probability value. For example, the optimal value of $p$ for image data in the input layers is 0.8 as proved in [119].

### 1.3.2.6 Fully connected layers

After all the preprocessing, a flattening function is added, thus image data is transformed into a feature vector representation for feature-learning process, which is taken care of by fully connected layers. Those extracted features are fed to FC layers to classify images as
a last part of CNNs. FC layers is just another name for regular neural networks like MLP, ANNs and sometimes dense layers as described in the introduction of this chapter.

It is only normal to ask the following question: why not use MLP directly for image classification? Well, first of all, to feed an image directly to a MLP, it needs to be flattened which breaks any information’s relation that may be present in the image. Moreover, MLPs perform poorly compared to CNNs for image classification.

The basic components of FC layers are as follows:
- **Input layer**: it contains the flattened feature vector extracted from previous layers.
- **Weights**: they represent the percentage of importance of a node in a layer, hence the final output prediction.
- **Hidden layers**: it’s a type of layers that their inputs and outputs are supposed to be uncontrollable, they contain a number of nodes called neurons stacked on top of each other.
- **Output layer**: after that data is fed to the input and passed through hidden layers, the output layer decides which real value to output in case of regression or a set of probabilities in case of classification.

We discussed MLPs deeply in the introduction, so for further information see section 1.3 of this chapter. Fig.1.1 in the mentioned section shows an example of fully connected layers architecture.

### 1.4 Deep Learning Applications in Medical Image Analysis

Medical images seem to be different from other types of images in different ways [21]. Moreover, when it comes to image processing, medical images usually illustrate sensitive information, which is mostly relied on in clinical interpretation and diagnosis. The issue with this information is that they are easily damaged or can be lost during the processing operation [74], or even during acquisition.

For human body organs acquisition, there are various options, depending on the type of the acquired organ and the quality of acquisition. The most used modalities in medical practice cases are; Computer Tomography (CT) [65], Positron Emission Tomography (PET) [153], and Magnetic Resonance Imaging (MRI) [107]. Each one of them has its own way of obtaining the organ information. For instance, MRI scans use magnetic wave and CT scans use X-rays . . .

In the beginning of medical image analysis, researchers used to use sequential application of low-level pixel processing (basic shape detection filters, region growing) and mathematical modeling (fitting lines, circles and ellipses) to construct a mixture of rule-based systems that interpreted particular tasks. Expert systems were called the good old-fashioned artificial
intelligence and they were very similar, in some ways, to rule-based image processing systems. Thereafter, the introduction of supervised techniques came to be increasingly popular in the medical image analysis domain. Therefore, image segmentation through active shape models, atlas models, feature extraction methods and statistical classifiers for diagnosis or objects detection, they helped in shifting from human designed systems, to systems trained by computers with a very little intervention from humans, considered in order to obtain handcrafted features.

The added value brought by deep learning into medical image analysis domain, as well to other domains, was the automation of the feature extraction phase. This concept of combining feature extraction layers (extract high levels of abstraction) with feature learning ones (previous layers make it easy for these layers to perform inference) is integrated in many deep learning models, which is basically the key to the strength of such architectures.

There are different applications of deep learning in the medical imaging field, from detection through segmentation into classification. In the following we will present each application separately. Note that those applications are part of what is called Computer Aided Diagnosis (CAD) system. A CAD system is a tool designated to help medical doctors/radiologists in order to provide an accurate diagnosis decision. The efficiency of a CAD system is measured by its accuracy, speed and automation level.

For example, the lifecycle of a CAD system for lung cancer diagnosis relies on these four major steps: segmentation of the lung fields to eliminate any useless area and focus on the area of study (lung parenchyma), detection of lung nodules inside the lung parenchyma, segmentation of the detected nodules, and then classification of segmented nodules into two classes: malignant and benign.

### 1.4.1 Detection

In image processing, object detection is one of the major steps. It consists of detecting instances of semantic objects of a given class. In general, object detection can have a general meaning of image classification of certain images that may or may not contain an object. It can be confusing some times to distinguish between object classification, object localization and object detection. In object classification, we assign a class label to an image, whereas object localization consists of surrounding an object inside the image by a bounding box. Object detection is a combination of both classification and localization, it draws a bounding box around the object and assign it a class label. The performance of a model for object detection is evaluated using the precision and recall for the known objects inside the image across each of the best matching bounding boxes. Now that all made clear, we will present in the following, some recent state-of-the-art deep learning models for image detection and their
application in the medical field. Object detection in the medical field refers to the detection of lesions and organs in medical images.

There are numerous object detection architectures in deep learning that are very popular, such as R-CNN, fast R-CNN, faster R-CNN, YOLO in all its versions etc.

R-CNN is a region-based CNN that was proposed by Ross Girshick et al. [36] in 2014. It was one of the first large and successful applications of CNNs in the object detection domain. R-CNN achieved state-of-the-art performance in both VOC-2012 and the 200-class ILSVRC-2013 data sets [78]. R-CNN consists of basic three building blocks; region proposal block, which generates and extracts candidate bounding boxes, meaning proposal of category independent regions. Thereafter, feature extraction from each candidate region using a deep CNN based on AlexNet architecture [68]. Finally, the classifier block that consists of classifying features given the classes at hand.

In [78] R-CNN was applied to prostate cancer detection based on Gleason grading using histological images. Wenyuan Li et al. demonstrated with the utilization of R-CNN for multi-task prediction they managed to provide complementary contextual information which led to better performance compared to single task model. Authors claimed that with this model, they achieved state-of-the-art performance in epithelial cells detection with 99.07% accuracy. Fast R-CNN is a variant of R-CNN in terms of speed. In 2015, Girshick et al. [35] proposed a faster version of R-CNN that addresses the issue of involving three separate models which is computationally expensive, however, in [35] authors proposed a single model that performs object detection very fast and deals with the issues presented in [36]. Nevertheless, in 2016 Shaoqing Ren et al. [116] proposed further improvements of the Fast R-CNN architecture in terms of training speed and detection accuracy. The proposed architecture is designed to propose and refine region proposals. Thereafter, these regions are used as input to Fast R-CNN in a single model design. In both R-CNN and fast R-CNN, the selective search is used to find the region proposals, which is time consuming and computationally expensive. Nonetheless, in [116] authors got rid of the selective search and replaced it by letting the network learn the region proposals itself, they called it Faster R-CNN.

In [160], SuYang et al. performed microscopic cell detection based on Faster R-CNN architecture. Authors showed in their paper, that with the use of Faster R-CNN in microscopic cell detection they manage to detect almost every microscopic cell with a much faster and accurate performance.

The R-CNN family is generally accurate, yet the You Only Look Once (YOLO) models are much faster than R-CNN even achieving real-time object detection accurately. Opposed to prior works in object detection, YOLO [112] does not repurpose classifiers to perform object detection but instead, it frames object detection as a regression problem. In a single
pass, the model performs object detection and classification via bounding boxes and class probabilities respectively, directly from the input image. The YOLO architecture can process images in real time at 45 frames per second, and still achieve promising results very fast. The base YOLO model suffers from localization errors compared to other object detection methods, but it learns general representations of the objects and outperforms R-CNN when dealing with different kinds of images such as artwork images.

There are various variants of YOLO that provide even more accurate and faster object detection. For instance, YOLOv2 [113], improves the first version by adding batch normalization and can deal with higher resolution input images. Besides, the choice of bounding boxes is preprocessed using k-means during training. Further enhancements were added to the YOLOv2 and presented by Redmon et al. in [114]. YOLOv3 has known very basic improvements in terms of architectural design and depth without compromising the accuracy and time performance. Researchers are making improvements of YOLO (in terms of architectural design, time performance and accuracy) every now and then, for instance we are now talking about YOLOv5.

In fact, YOLOv3 was applied to kidney detection in CT scans [73]. It showed very promising results by achieving 0.851 dice score in 2D CT scans and 0.742 in 3D.

There are many applications of YOLO family in the medical domain such as Breast masses detection and classification using YOLO [1], lung nodule detection in CT scans using YOLO [9] then detection and classification of cholelithiasis and gallstones in CT images [109].

1.4.2 Segmentation

The next step into image processing is image segmentation. It is a challenging image processing task, which requires computational friendly and accurate algorithms. In each general introduction of each chapter from part II, we describe in details image segmentation and we mentioned few advanced deep learning architectures for that purpose. Medical image segmentation [129] helps in highlighting and analyzing a certain region of interest such as lung tissues, spleen, brain etc. there are many applications in the medical image segmentation from which we can refer to brain tumor segmentation with boundary extraction in MRI slices, cancer detection in lung CT scans, segmentation of affected area in chest X-rays and so on. As there is a shortage of experts in such domains, a number of well-designed algorithms in terms of speed and accuracy are proposed in the literature to serve the purpose of diagnosis [103].

Here we are mentioning some application of those architectures in the medical domain. For instance, in [43] authors adopted a novel two-pathway CNN based architecture to perform
brain tumor semantic segmentation. This architecture can capture local features to learn about local details of the brain as well as the large contextual features. In addition to the architectural contribution, authors proposed a two-phase training strategy which seems to deal in an efficient way with imbalanced labels distributions. In another work related to the medical domain, Baumgartner et al. [8] proposed a 3D CNN based architecture for cardiac MRI images segmentation into left and right ventricular cavities and myocardium. In [150] Wang et al. performed a pneumothorax segmentation in X-ray images adopting a spatial and channel Squeeze-Excitation CNN based architecture. In [43] authors designed CNN architecture for brain tissue segmentation in MRI images. Based on FCN architecture, Zhang et al. [169] proposed a liver segmentation in CT scans. Christ et al. [25] designed a two cascaded FCN architecture for liver segmentation, thereafter, the final segmentation result is provided using a dense 3D conditional random field.

1.4.3 Classification

The final step into medical image processing is classification. It is a mandatory task for delivering a precise diagnosis. The classification accuracy depends on the quality of all the previous steps in an image processing journey, besides the utilized classification models.

Image classification is a supervised learning problem used to define a set of target labels then trained a model to recognize them in new input images.

In each general introduction of each chapter from part III, we described in detail some advanced deep learning architectures for image classification. Hence, in this section we are only going to present some of their applications in the medical field.

There are two strategies of transfer learning, first one uses a pre-trained network as a feature extractor while the second fine-tunes a pre-trained model on new data. Antony et al. [20] used transfer learning to train a CNN as a fine-tuning model on medical data and achieved 57.6% accuracy in grade assessment of knee osteoarthritis compared to feature extractor strategy 53.4%. Whilst, Kim et al. [3] showed the opposite, meaning that using a pre-trained CNN as a feature extractor outperformed fine-tuning strategy in cytopathology image classification by achieving 70.5% accuracy versus 69.1%. It is confusing when it comes to choosing between the two transfer learning strategies, however in [66, 29] authors showed that with the use of a pre-trained model as a feature extractor can not achieve better results than using fine-tuned pre-trained models. In fact, they fine-tuned a pre-trained version of the well-known Inception v3 deep architecture on medical data and nearly reached human expert performance.

In [40], Shen et al. performed a lung nodule classification relying on a multi-scale CNN, authors used three different CNNs where each one deals with a different scale lung nodule
patch. In the end, the outputs of the three CNNs are combined to form the final feature vector. A similar approach was used by Kawahara and Hamarneh [131] for skin lesions classification.

Processing natural images often tend to be in a 2D way, and computer vision networks only leverage those kinds of images. However, in medical image processing, images tend to have a 3D format. Many works have tried to integrate 3D in an efficient way. For instance, Nie, et al. [64] trained a 3D CNN for 3D MRI images to classify high grade gliomas.

Furthermore, Setio et al. [128] performed a lung nodule classification (chest region of interest as nodule or non-nodule) using a 3D multi-stream CNN in chest CT scans.

1.5 Conclusion

In this chapter, we presented a deep learning overview and a detailed CNN architecture. We also presented some advanced DL architectures for medical image analysis. In the rest of this thesis, we concentrate on presenting comparative studies focusing on enhancements on DL architectures, and introducing new DL methods for medical image analysis.
Part II

Medical Image Segmentation
Introduction

Image segmentation is a technique used in image processing for the purpose of partitioning an image into multiple parts or regions based on their pixel’s characteristics. Image segmentation is simply used for the purpose of reducing the complexity of the image for making further image processing easier. Image segmentation makes all the elements or pixels that belong to the same category get assigned their proper label. In other words, instead of processing the whole image, segmenting the region of interest and then analyze it seems much simpler, the principle of “divide to conquer”. The resulted image is then processed separately instead of the whole image, which makes the inference time reduced. In a work of ours [136] chapter 2 from part II, we used image segmentation in lung CT scans in order to separate the lung area from other parts that we are not interested in, then used the region of interest for lung nodule detection. Image segmentation can be categorized into two categories: semantic segmentation and instance segmentation.

Semantic segmentation identifies all objects of the same type as one class, while instance segmentation identifies similar objects each one with its own label. To elaborate more, let’s consider an image that contains 5 people. Semantic segmentation classifies all the five people as one instance with the background is identified as another class, while instance segmentation will segment each one of these people individually.

There are different techniques in the literature to perform image segmentation, from which we can mention a few:

Threshold-based image segmentation, edge-based image segmentation and Artificial Neural Networks based image segmentation. In the segmentation part of our thesis, we perform ANN based image segmentation.

Since deep learning was born, several deep-learning-based architectures for image segmentation were born as well, most of them were based on CNNs. Lately, performing image segmentation using CNNs became widespread across several businesses and industries due to their effectiveness in dealing with image data sets in general. Because of their ability of extracting relevant features, CNNs are known to achieve very promising results in healthcare in general. For instance, in medical image segmentation, CNNs have outperformed
significantly many conventional machine learning methods, hence achieved state of the art results [94]. They became the most popular choice for multiple medical image processing fields, we mention few of them such as; brain MRI [97], lung nodule [27], spleen [120] and cardiac medical imaging issues [33].

Since our contributions in this part rely on FCN like architecture, Auto-Encoder based architecture, Multi-Scale like architecture and Attention based architecture, we are presenting in this general introduction some of the advanced CNN architectures that are very popular in image segmentation based on their model architecture.

**Fully Convolutional Networks (FCN)**

Based on an architecture fully consisting of convolutional operations called Fully Convolutional Networks, Long et al. [90] introduced one of the first deep learning architectures that are suitable for semantic segmentation. The way a FCN is built enables it to deal with arbitrary sized images in order to generate a corresponding segmentation map with the same size. The authors made a modification on the well-known VGG16 and GoogleNet to deal with the problem of non-fixed sized images by replacing fully connected layers by fully convolutional layers. Fully connected layers give classification scores as output, while fully convolutional layers generate spatial segmentation maps. Moreover, authors used skip connections which takes feature maps from the final layers of the model then they are up-sampled and combined with earlier layers of the model in order to keep as much as possible of features from the input image and help getting accurate segmentation result. FCN has showed great performance in several data sets which demonstrates that deep neural networks are trainable for semantic segmentation in an end-to-end way on data sets containing variable sized images. This work is considered a milestone in image segmentation. Nevertheless, FCN has some limitations that made it a little bit far from satisfactory: it cannot be transferable to 3D image data sets, it lacks speed for real time inference and it does not take into consideration the global context information in a robust way. However, many works have attempted to address those limitations. From which we can mention for instance ParseNet proposed by Liu et al. [87], that consists of overcoming the issue with global context information, simply by adding average features to capture mode general features at each location. In a layer, the feature map is pooled over the whole input resulting in a context vector. This context vector normalized and unpooled to generate new feature maps having the same size as the initial ones. Then, these feature maps are concatenated. FCNs have been performed in many works such as skin lesion segmentation [148], iris segmentation [80], instance aware segmentation [163] and brain tumor segmentation [86].
Convolutional models with graphical design

One of the limitations of FCNs we discussed above is the lack of capturing useful global context features. In order to handle this kind of issues, a lot of approaches integrated graphical models into deep learning architectures, from which we bring up Conditional Random Fields (CRFs) and Markov Random Fields (MRFs). In [14], Chen et al. introduced a novel DL architecture based of CNNs combined with fully connected CRFs. Authors showed that this model is capable of localizing boundaries of each segment at a higher accuracy rate. In another work related to combining CNNs and CRFs, Schwing and Urtasun [127] presented uses both methods jointly in order to perform semantic image segmentation in an accurate way, which showed encouraging performance results in several image data sets. In another work [175], Zheng et al. integrated CRFs in a CNN architecture to perform semantic segmentation. In [82], Lin et al. introduced a semantic segmentation algorithm based on contextual deep CRFs. To elaborate more, they made use of the contextual information to explore patch-patch context and patch-background context in order to improve the algorithm of semantic segmentation. Last but not least, Liu et al. [88] proposed a novel semantic segmentation algorithm that integrates rich features into MRFs including high-order relations and label contexts.

Encoder-Decoder based models

In this thesis, we managed to perform image segmentation in the medical field. However, most of state-of-the-art DL architecture that were introduced in the literature were first applied on image data sets other than those of the medical field. Thus, in this chapter we group these works that we are mentioning after into two categories, encoder-decoder model for segmentation in general, and then for medical image segmentation.

General segmentation with encoder-decoder models

In [105], authors performed a semantic segmentation using a two parts architecture, first part consists of an encoder based on adopted layers from the well-known VGG-16-layer model, then second part that integrates deconvolution layers in a consecutive way with upsampling (unpooling) layer to generate a map of pixel-wise class probabilities called segmentation mask. In another work, Badrinarayanan et al. [5] proposed a novel encoder-decoder architecture named SegNet. Similar to the deconvolution network beforementioned, it consists of the early 13 layers of the VGG-16 network followed by a pixel-wise classification layer to predict each pixel’s class. Another popular encoder-decoder based work called
high resolution network HRNet [164] consists of maintaining high resolution representations through the encoder process, then repeatedly exchanging the information across resolutions. Many works adopted the encoder-decoder strategy to perform image segmentation, such as W-Net [13], Stacked Deconvolutional Network SDN [156], and LinkNet [18]. However, Encoder-Decoder architectures suffer from the loss of fine-grained information of the image, and that is due to the loss of high-resolution representations over the encoding process, HRNet is one of the works that addresses such vulnerability.

Medical and biomedical image segmentation using encoder-decoder models

There are many medical image segmentation works that were inspired by encoder-decoder and FCN models, such as U-Net [118] and V-Net [93]. The U-Net architecture was performed on biological microscopy images. It consists of using data augmentation to learn from the very few annotated images in a very efficient manner. This network, consists of two parts, the first part called the contracting path which consists of high-level feature extraction, then the expanding path that relies on the extracted feature maps, in addition to copying and cropping from the corresponding layer in the contracting path to avoid losing pattern information and to perfectly recover the required features of the segmentation mask. On the other hand, V-Net is a 3D medical image segmentation algorithm proposed by Milletari et al. Authors introduced a new objective function based on Dice Coefficient enabling the model to deal with situations in which there is a strong imbalance between the number of voxels in the foreground and background. Many works adopted U-Net and V-Net architectures, from which we mention the following: Zhou et al. [22] developed a nested U-Net architecture. Cicek [176] proposed a U-Net architecture for 3D images. Zhang et al. [172] developed a road segmentation/extraction U-Net-based algorithm. The same goes for the V-Net architecture, Progressive Dense V-net (PDV-Net) et al. for fast and automatic segmentation of pulmonary lobes from chest CT images, and the 3D-CNN encoder for lesion segmentation [12].

Multi-scale and pyramid network based models

It is an old idea in image processing, it has been adopted in many neural network architectures. One of the most popular models of this family is the Feature Pyramid Network FPN [84] proposed by Lin et al. It was mainly built for object detection and then was extended to image segmentation. A combination of the multi-scale and pyramidal hierarchy of CNNs was adopted to build feature pyramids, to merge low- and high-resolution features.
FPN is composed of top-down, a bottom-up pathways and lateral connections. The generated feature maps are then fed to a $3 \times 3$ convolution to generate the output of each stage. Finally, each stage of the top-down pathway produces a prediction to separately detect an object. On the other hand, for image segmentation, they used two-multilayer perceptrons.

To better learn the global context features in a scene, Zhao et al. [23] proposed a pyramid scene parsing network which is a multi-scale network. With the use of a ResNet as a feature extractor, multiple patterns are extracted from the input image in a dilated network. Thereafter, those features are fed to a pyramid pooling module in order to identify different patterns of different scales. These feature maps go through four different pooling scales, each one belongs to a corresponding pyramid level, then followed by a $1 \times 1$ convolutional layer for dimensionality reduction purposes. Next step, is up-sampling the outputs of the pyramid levels and concatenated with their corresponding initial feature maps to acquire both global and local characteristics. Finally, generating pixel-wise prediction based on a convolutional layer. There are various models in the literature that adopt a multi-scale analysis strategy for image segmentation such as Adaptive Pyramid Context Network [44], Context Contrasted Network [81], Salient Object Segmentation [15] and so on.

**Semantic segmentation using attention-based models**

Attention mechanism have shown very effective in many works over the last few years, it suits the semantic segmentation task very well given its nature in focusing on each object separately. Chen et al. [57] introduced in their paper an attention mechanism that joins multi-scale features at each pixel location. Authors manage to jointly merge a powerful segmentation model with the attention mechanism in a multi-scale architecture. In another work, Huang et al. [76] proposed an approach for semantic segmentation that is based on reverse attention mechanism. Opposite to attention mechanism, reverse attention mechanism captures the features that are not in the scope of the model’s aim. In other words, it captures features that are not associated with the targeted class. The RAN model captures the features mentioned above, in addition to attention mechanism’s features. It is a three-branch network that performs simultaneously both feature extraction attention mechanisms (direct and reverse), in order to contain high level features. In [31], Li et al. proposed a Pyramid Attention Model dedicated for semantic segmentation and it gathers between spatial pyramid and attention mechanism concepts to exploit effect of global contextual information for a better extraction of dense features in order to efficiently label pixels. In a recent work [165], Fu et al. developed a dual attention network for scene segmentation, that can generate contextual dependencies based on the attention mechanism. Furthermore, they adopted a
dilated FCN for inter-dependencies in both spatial and channel dimensions in addition to those extracted contextual dependencies.

Several other attention-based works have been adopted for semantic segmentation, such as OCNet [58], Criss-Cross Attention Network [115], Discriminative Feature Network DFN [161] and end-to-end instance segmentation with recurrent attention [117].
Chapter 2

Lung CT Image Segmentation Using Deep Neural Networks

Contents

2.1 Overview ........................................... 42
2.2 Methods ............................................. 43
2.3 Results and discussion ............................. 45
2.4 Conclusion .......................................... 47

2.1 Overview

Lung CT image segmentation is a prerequisite step for any kind of lung image analysis, it is a necessary step to obtain a satisfactory result in CT image analysis tasks, such as lung cancer detection. In order to get an accurate image analysis, it is important to properly provide a lung area segmentation.

Lung cancer is one of the most common cancers in the world, it is a dangerous lung disease that causes more than one million death each year [117]. Lung cancer is a malignant lung nodule that is described by uncontrollable growth. To prevent from death by lung cancer, early detection remains the best way to increase the patient’s survival rate.

For lung cancer diagnosis and detection, Computed Tomography (CT) is an efficient medical screening test used for that purpose. Nevertheless, physicians in many cases find it difficult to obtain an accurate diagnosis without the assistance of Computer Aided Diagnosis (CAD) Systems. CAD is an additional tool known for its help in providing assistance to
physicians in order to obtain accurate diagnosis. In today’s medical imaging, it is mandatory to obtain assistance from CAD systems in order to provide an efficient medical diagnosis.

The success of a CAD system resides on providing an accurate segmentation of the organ targeted, it is a mandatory initial step for an efficient lung CT image analysis.

Nevertheless, for a better lung segmentation, the lung parenchyma needs to be separated from the bronchus region because they are often confused with the lung tissue. Furthermore, in the case of abnormal lung parenchyma, it is a challenging task to include lung nodules and blood vessels in the segmentation along with the lung parenchyma, thus it is a complicated problem to exactly provide a thorough lung segmentation.

In this chapter, we cover the presentation of lung segmentation using a commonly used architecture for image segmentation based on deep learning called U-net [118]. This work is a first step into lung cancer detection, by erasing irrelevant information acquired in lung CT scans. Our network achieves an accurate segmentation, 0.9502 Dice coefficient score, based on a data set that contains few hundreds of manually prepared training lung images.

The rest of this chapter is organized as follows: in next section we provide an overview of the U-Net architecture. Then in section that follows we present the resultant segmentation accuracy using LIDC data set, then concluding remarks.

### 2.2 Methods

In very few years, deep learning has dramatically improved over the state-of-the-art in different domains. The era of deep learning began with the proposition of a new algorithm with a new learning strategy by Hinton et al. [51]. Deep learning in general and CNNs in particular have known an exceptional raise to be the most approved algorithms for a variety of pattern recognition tasks. Until today, deep learning algorithms are still winning many international competitions in speech and image recognition [126].

As for image classification, CNNs has had a tremendous success in image segmentation problems. For example, in 2015 Long et al. [90] proposed a new architecture named Fully Convolutional Networks (FCN), that made CNNs very popular for dense predictions without any fully connected layers. It allowed to generate segmentation masks for any type of images in a much faster way compared to its classical competitors for image segmentation.

In FCN architecture, performing image segmentation without fully connected layers was not the only challenge, but also the pooling layers. Pooling layers reduce the size of their inputs. To deal with this issue, up-sampling layers were adopted. It consists of performing the reverse task as pooling operation. Hence, in this encoder-decoder architecture, pooling layers were responsible for reducing the spatial dimensions of objects in the encoder part.
On the other part, the decoder, the up-sampling operation is responsible for recovering the objects relevant details.

Encoder-decoder architecture type has many variants such as FCN and U-Net. In this work we adopt the U-Net architecture, one of the most popular architectures for medical image classification.

In 2015, Ronneberger et al. [118] proposed the U-Net architecture (see Fig. 2.1) which is based on FCN. U-Net’s basic components can be viewed as an association of convolution layers in the encoder part which is named the contracting path, then deconvolution layers in the decoder part, called expansive path. The contracting path is formed as a standard CNN, it consists of convolution layers with Rectified Linear Units (2.1) (ReLU see Fig. 2.2) as activation function, then followed by a max-pooling layers. On the expansive path, the resultant feature maps are fed to up-sampling layers followed by up-convolution and convolution layers with ReLU activation function. As consequence of the loss of border pixels at every convolution operation, it is mandatory to crop the missing parts from the corresponding feature maps in the extracting path and concatenate it with its parallel correspondent in the expansive path.

\[
    f(x) = \max(0, x) \tag{2.1}
\]

In the training phase, the U-Net architecture uses input images and their corresponding masks as outputs. While in the test phase, input images are fed the network to generate the corresponding mask as output, Fig. 2.3 show the input and the output examples. The output mask is then applied to its corresponding input image to perform the segmentation of the area of interest in the image, i.e. the lung parenchyma in our data set.

To training our network, we use the Lung Image Database Consortium image collection (LIDC) for lung CT scans [4]. It consists of diagnostic and lung cancer screening thoracic CT scans with marked-up annotations of lesions. It is a web-accessible resource mad available for the development of CAD systems for lung cancer diagnosis.

The data set only contains CT scans of the whole lung area; thus, a data preparation was needed to obtain a suitable data set for our network which is composed of images and their corresponding output masks. The ground truth of lung area was provided manually. We performed a necessary pre-processing step that consist of images cropping to remove any irrelevant information that doesn’t belong to the area of study.
2.3 Results and discussion

We performed our experimentation in an environment composed of Keras and NVIDIA GTX1050. Keras is high-level API made for especially for neural networks. It is written in python and runs on top of either Tensorflow or Theano with the possibility to run on CPU or GPU. Therefore, our experimentation took place on a Tensorflow backend and a GPU environment with graphics processor NVIDIA GTX 1050 equipped with 640 CUDA cores and 4Gb of memory in order to make profit of its maximum computational speed.

We set the network parameters to:

- Batch size: 32.
- Number of epochs: 50.

For evaluation purposes of our network, we use dice coefficient score DSC as a similarity metric considering that it is one of the most used similarity measurements in image segmentation; it is calculated using the following formula:

![Fig. 2.1. The U-Net architecture.](image-url)
2.3 Results an discussion

**Fig. 2.2.** Rectified Linear Units (ReLU), presented in (2.1).

**Fig. 2.3.** The way that U-net architecture works is that it takes the input image to generate the corresponding map for segmentation.

\[
DSC = 2 \frac{||S \cup T||}{||S \oplus T||}
\]  

(2.2)

The \(S\) stands for the lung parenchyma area obtained using our network, and \(T\) represents the ground truth provided by manual segmentation. Using our network, we’ve reached an average DSC of 0.9502. **Fig.2.4** presents experimentation results that show resultant segmentation performed using our network. The results are presented under five columns, first column from the left represents the original input image, the second column going to the right is the ground truth of lung parenchyma manual segmentation masks, then the next column presents the resulted segmentation maps generated using our network, next column represents the desired segmentation generated using manual segmented masks, and finally the last column is the segmentation result provided by the generated segmentation maps using our network applied on the input lung image.
2.4 Conclusion

We can see that in the 5th column, the lung segmentation obtained using the U-Net architecture does not contain any parts of the trachea nor the bronchos regions and it does not eliminate any lesions such as lung nodules and blood vessels that are important for upcoming tasks of diagnosis, this result shows the accuracy of our network.

2.4 Conclusion

In this chapter, we provided a study about lung parenchyma segmentation using one of the most efficient deep learning segmentation architectures, called U-Net. The efficiency of such network resides on the good quality of the segmentation result by using only few hundreds of images in the data set. We obtained using the U-Net architecture, 0.9502 dice coefficient index score, which is a high score for such kind of tasks. The advantage of this approach is that it is uniform and can be applied to a wide area of different medical image segmentation tasks. This work is an introduction to understand how deep learning architectures can contribute to the medical image segmentation domain and a first step to propose new deep learning architectures tailored for medical image segmentation. In the next chapter, we are proposing a novel architecture based on deep learning to perform brain tumor segmentation.
Chapter 3

Multi-Scale ConvLSTM Attention-based Brain Tumor Segmentation

Contents

3.1 Overview .................................................. 48
3.2 Related work ............................................... 50
3.3 Method .................................................... 52
3.4 Experiments and results ................................. 56
3.5 Conclusion ................................................ 57

3.1 Overview

In the previous chapter, we performed lung ct image segmentation using a deep learning architecture that is known to work efficiently with small amounts of data. In this chapter, we introduce a novel deep learning architecture to perform medical image segmentation and we compare its performance to the one used in the previous chapter besides other related methods. Our goal is to show that with our novel method that adopts robust feature extraction methods surpasses the methods we used in our previous chapter that uses conventional feature extraction methods.

Nowadays, the use of CNNs is widespread across industries and businesses. In healthcare, CNNs achieve very promising results due to their robust feature extraction capabilities. For example, in medical image segmentation, they have achieved state-of-the-art performance [95] with a significant margin compared to conventional machine learning models, which makes them the most popular choice in different medical imaging fields. They also dominate the
health informatics literature on brain [97], lung nodule [42], spleen [121], and cardiac [34] medical imaging issues, to mention a few.

In this chapter, we perform brain tumor semantic segmentation using a novel deep learning architecture. Brain tumors are considered one of the deadliest cancers in the world. There are various brain tumor types, but gliomas are the most common ones among adults. Furthermore, gliomas can be present with different degrees of aggressiveness with an average survival time for patients diagnosed with glioma lesser than 14 months [146]. Therefore, time is a critical factor for doctors to act regarding gliomas. To diagnose a brain tumor, there are different types of medical image acquisition involved, such as MRI, CT scans and X-Ray, each having its pros and cons. For example, CT scans have the advantage of speed of tissue acquisition at the cost of lower quality of tissue contrast and higher radiation risk. On the other hand, MRIs are slow compared to CT scans but they are best suitable for capturing abnormal tissues with more details due to their accuracy in acquiring different types of contrasts. After the acquisition of the brain region, radiologists perform a manual segmentation of brain tumors from MRI images, which is time-consuming. Therefore, designing an automatic brain tumor segmentation is mostly desirable.

We propose a novel deep neural network, called Multi-Scale ConvLSTM Attention Neural Network (MSConvLSTM-Att), to automatize brain tumor semantic segmentation. Our architecture is multi-scale-attention based with each level using Convolutional Long Short Term Memory (ConvLSTM) [157], Squeeze and Excitation-inception (SE-inception) [142] and Squeeze and Excitation-Residual-Network (SE-ResNet) [142].

The use of such multi-scale architecture, which is composed of multiple stages, is to generate multiple versions of the same image with different resolutions, each containing diverse semantics. The first low-level stage serves to model the spatially sequential relationship between different parts of each MRI modality (FLAIR, T1w,T1gd,T2w) 1, while the next stage manages the extraction of local features in addition to decreasing the size of the images for computational optimization. Finally, the third high-level stage captures the global representations. Thereafter, at each level, we introduce a stack of attention modules to gradually emphasize the regions that contain a large number of semantic features.

The integration of attention mechanism in the image segmentation of natural scenes has been widely adopted [77, 31, 16, 173]. However, in medical imaging, the inclusion of attention mechanism is rare [152, 75, 125, 104]. For this reason, we investigate the impact of a simple attention module in boosting the performance of standard deep networks for brain tumor semantic segmentation. Experimental results show that our proposed method improves

1https://case.edu/med/neurology/NR/MRI Basics.htm
the segmentation performance by modeling a combination of rich contextual features with local features.

The results shown later on this work prove that our model performs semantic brain tumor segmentation effectively compared to standard U-Net, Attention U-Net and Fully Connected Network (FCN), by reaching 79.78 Dice score index using our model compared to 78.61, 73.65 and 72.89 using Att-UNet, UNet and FCN respectively.

The remainder of this chapter is organized as follows. Next section presents related works. In the section that follows, we introduce our proposed method in detail. Thereafter, we present and discuss the obtained results. Finally, we conclude our work in the last section.

3.2 Related work

Most of the state-of-the-art deep learning architectures used for automatic medical image segmentation are inspired from Fully Convolutional Networks (FCN) [90] or U-Net [118]. Many variants of these architectures have been proposed to perform semantic segmentation in different application domains [24, 79, 47].

FCN is an architecture in which fully connected layers are replaced by deconvolution layers to generate segmentation masks [90]. Jesson et al. [60] proposed a variant of the standard FCN with a multi-scale loss function. With this approach it is possible to model the context in both the input and output domains. A limitation of this approach is that FCN is not able not explicitly model the context in the label domain. Compared to U-Net, FCN does not use skip connections between the contracting (i.e feature extraction path) and the expanding paths (i.e data reconstruction path).

The U-Net architecture was introduced by Ronneberger et al. in 2015 [118]. It overcomes the limitations of FCN by including features from the contracting path. In order to obtain the missing feature-contexts, multi-scale features are concatenated in a mirroring way. Many works have adopted this architecture to perform medical image segmentation over different parts of the human body. In a previous work of ours [136], we also adopted the U-Net architecture to perform lung CT image segmentation.

A limitation of both FCN and U-Net is that they both do not perform very well in multi-class segmentation tasks [60]. To overcome this issue, cascaded architectures can be used. They have the beneficial effect of decomposing a multi-class segmentation problem into multiple binary segmentation problems. This approach is also used in various medical image segmentation works. For example, Chen et al. [17] adopted a cascaded classifier to perform a multi-class segmentation. Furthermore, in [85] authors proposed a cascaded architecture to merge different feature extraction methods. Nonetheless, these models still face a problem
of focusing on pixel level classification while ignoring adjacent pixels’ connections. To overcome this issue, Generative models were adopted. A widely used variant of generative models is Generative Adversarial Network (GAN) [39]. GANs are employed for semantic segmentation in the following way: a convolutional semantic segmentation network is trained along with an adversarial network to discriminate segmentation maps [139]. That is, two models are trained; the first captures data distribution, while the second is used for a discriminative purpose.

To capture sequence patterns in medical imaging, Recurrent Neural Networks (RNNs) are typically used as they are well suited for handling sequential data. Specifically in medical image segmentation, RNNs are used to keep track of features in previous image slices in order to better generate the corresponding segmentation maps. There are various RNN architectures mentioned in the literature, and amongst them Gated Recurrent Units (GRU) [19] and Long-Short Term Memory (LSTM) [55] are likely the most robust and widely used. GRU is memory efficient, nonetheless not very suitable for keeping track of long-term features. LSTM is better adapted to such tasks due to the forget gate that preserves features from previous sequences to use in upcoming sequences.

[2, 69, 174] are some examples of employing RNNs for performing image segmentation for sclerosis lesions and brain tumors respectively.

In the last few years, a new concept called attention mechanism was introduced into computer vision tasks. Attention mechanism was introduced first in neural machine translation [6] to help remember long range context from long source sentences. The added value brought by attention modules is the creation of shortcuts between the input sentence and the context vector. Attention in deep learning can be interpreted as a vector of weights that represent the importance of an element within a context. The attention vector is used to estimate how strongly is an element related to other elements (elements in this context are image pixels), it takes the sum of these elements’ values weighted by the attention vector as the approximation of the target context.

The success of the attention mechanism for neural machine translation has encouraged its application to computer vision immediately [159]. In medical image segmentation, the attention mechanism was adopted in many works and various variants of attention modules have been introduced.

In [122], authors propose a combination of FCN with a Squeeze and Excitation (SE) attention-based module to perform whole-brain and whole-body segmentation. They integrate the SE block in three ways: channel SE (cSE), spatial SE (sSE) and concurrent spatial-channel SE (csSE). In [152], Wang et. al. perform prostate segmentation in ultrasound images using deep attentional features. They use an attention module to extract refined
features at each layer, eliminate nonprostate noise and focus on more prostate details at deep layers. Furthermore, Li et. al. propose an auto-encoder CNN-based architecture, called hierarchical aggregation network (HAANet) [75], which combines the attention mechanism and hierarchical aggregation to perform 3D left atrial segmentation. In another work, Oktay et al. propose an attention U-net [106] which extends the U-Net architecture by incorporating an attention gate in the expanding path in order to accurately segment the pancreas area.

3.3 Method

In this section, we describe our proposed architecture for brain tumor segmentation. Our method combines different techniques in order to extract relevant features and keep track of them during the entire process of segmentation.

We combine Inception, ResNet and Squeeze-Excitation blocks in one part of our architecture for relevant feature extraction, and attention modules in another part to perform brain tumor segmentation. The combination of Inception, ResNet and Squeeze-Excitation is known as the most successful architecture in the ImageNet challenge. With this combination, the team Trimps-Soushen achieved 2.99% error rate in object classification in the ImageNet challenge.\(^2\)

We first feed our network different modalities of brain MRI images (FLAIR, T1w, T1gd, T2w) to include various intensities and to better perform the semantic segmentation. Each modality is split into four patches, then for each modality, three scales of feature extraction are performed. The motivation behind this multi-scale mixture is to best separate each tumor label (enhancing tumor, tumor core, whole tumor and background).

At the first scale, ConvLSTM is used over each of the four patches to preserve the correlation among features. ConvLSTM are best suitable for catching spatiotemporal information without any much redundancy [157]. At the second scale, an SE-inception [46] module is used over the output of the first scale to extract low level features and decrease the computation cost. Fig.3.1 shows the inception module [143] and Fig.3.2 shows the SE-inception block.

At the third scale, we extract high level features by integrating an SEResNet module [61]. The use of such block increases computational complexity with a thin margin but in exchange of increasing the accuracy [61]. The ResNet block [46] and SE-ResNet are described in Fig.3.3 and Fig.3.4 respectively.

At each scale (different scales are highlighted by green color in Fig.3.6), we combine the four outputs to form what we call single-scale features as stated in Fig.3.6. These three

\(^2\)https://image-net.org/challenges/beyond ilsvrc
3.3 Method

Fig. 3.1. Inception block.

Fig. 3.2. SE-inception block.

single-scale features are then concatenated and convolved to form multi-scale features as mentioned in the same figure. We then take the multi-scale features and we combine them with each single-scale feature.

Fig. 3.3. ResNet block.

At this stage, our model holds general context feature-maps that contain different levels of features, from low to high level features. Thereafter, we add a convolution layer to refine these features. Furthermore, in order to explore more global contextual characteristics by building connections among features, we include attention mechanism in the form of a location-based attention module, we call it Spatial Attention Module (SAM). The attention mechanism is presented in the Fig.3.5.

In Fig.3.5, we assume the input to the SAM module is $V$, which is a 3D shaped input $(W,H,C)$, here $W$, $H$ and $C$ represent the width, height and depth respectively. In the red branch, we perform a convolution operation, resulting in a feature map $B_0$ with same width
and height but with depth equals to $C/8$. $B_0$ is then reshaped to $(W,H,C)$. The same operation is applied to the blue branch $B_1$. Thereafter, we perform a matrix multiplication and apply a softmax operation to calculate the spatial attention map following the formula in (3.1), where $S_{i,j}$ represents the impact of the pixel in the $i^{th}$ position on the pixel in the $j^{th}$ position.

$$S_{i,j} = \frac{\exp(B_0 \ast B_1)}{\sum_{i=1}^{W \times H} \exp(B_0 \ast B_1)} \quad (3.1)$$

The yellow branch performs a convolution and results in $B_2$ with the same shape as $V$. $B_2$ is then reshaped to $(C,W,H)$ then it is multiplied by the transpose of the spatial attention map $S$. Furthermore, the output $R$ is reshaped to $C \times (W \times H)$ and multiplied by a parameter $\lambda$ and then an element-wise sum with input $V$ is performed to obtain the output $O$ as expressed in (3.2).
\[ O = \lambda \sum_{i=1}^{W \times H} \exp(S_{i,j} \ast B_2) + V \] (3.2)

In (3.2), \( \lambda \) is initialized by 0 and gradually updated to give more weight to the spatial attention map, as adopted in [31].

At the last level, we perform a convolution operation to generate the final prediction map for each scale and then average all these maps to output the segmentation map. Fig.3.6 presents an overview of our proposed architecture.

Fig. 3.6. Overview of our proposed architecture.
3.4 Experiments and results

To evaluate our architecture, we are using BRATS’18 data set for brain tumor segmentation, provided in the Medical Segmentation Decathlon Challenge. This data set contains multimodal MRI data (FLAIR, T1w,T1gd,T2w). Furthermore, it contains 210 High Grade Glioma (HGG) scans and 75 Low Grade Glioma (LGG) scans. In this data set, the focus is mainly on the segmentation of different sub-regions of the glioma. First, the enhancing tumor (ET), the tumor core (TC) and finally the whole tumor (WT) as can be seen in Fig.3.7. Each one of these sub-regions have some specific characteristics regarding their intensities, hence different modalities are responsible for capturing different characteristics. For example, the ET is described by areas that are hyper-intense in T1gd. The appearance of the non-enhancing tumor (NET) (solid parts) and the necrotic (NCR) (fluid-filled) is represented by areas that show hypo-intensity in T1gd when compared to T1. The WT describes the whole disease and it contains the TC and the peritumoral edema (ED), which is characterized by hyper-intensity in the FLAIR modality. The provided labels in this data set are as follows: 1 for NCR and NET, 2 for ED, 3 for ET and finally 0 for other parts of the brain. The annotations were created by domain experts and approved by other domain experts as described in [7].

Given the presence of different features related to gliomas in different modalities, we feed the four modalities as input to our architecture, then we get the semantic segmentation that belongs to these inputs. The loss function we use is the dice loss optimized using the Adam optimizer. The learning rate is initially set to 0.001 and then multiplied by 0.5 after each 30 epochs. We used 500 epochs to train our network. Due to limitations in computational resources, we reduced the input size to 190 × 190 by cropping some of the background area and we only took from the 30th slice to the 120th given that most of the brain information is present in that interval. Furthermore, we normalize the inputs to have zero mean and unit standard deviation. In addition, given that each session of the notebook used for training has 12 hours lifetime, we use the following strategy to train our network. We save our model and its weights after each 50 epochs and we reload it and continue training with new data. For development, we shuffled and randomly split the images into training (225 patients), validation (30 patients), and test (30 patients). Experiments were performed in a server equipped with a single 12GB NVIDIA Tesla K80 GPU.

We compare our method with the standard UNet, standard FCN and the Attention U-Net architectures. And we evaluate their performance using the dice coefficient (DSC) as a comparison metric. Table 3.1 contains experimental results obtained using the different segmentation methods described above, and compared regarding their
DSC score. Our proposed architecture achieved the best score with 0.649, 0.881 and 0.865 in ET, WT, and TC respectively and a mean score of 0.798.

**Table 3.1.** Proposed method’s DSC score compared to those of U-Net, Att-UNet and FCN.

<table>
<thead>
<tr>
<th>Labels</th>
<th>ET</th>
<th>WT</th>
<th>CT</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-Net</td>
<td>0.563</td>
<td>0.848</td>
<td>0.797</td>
<td>0.736</td>
</tr>
<tr>
<td>Att-UNet</td>
<td>0.637</td>
<td>0.875</td>
<td>0.845</td>
<td>0.786</td>
</tr>
<tr>
<td>FCN</td>
<td>0.551</td>
<td>0.853</td>
<td>0.781</td>
<td>0.728</td>
</tr>
<tr>
<td>ours</td>
<td>0.649</td>
<td>0.881</td>
<td>0.865</td>
<td>0.798</td>
</tr>
</tbody>
</table>

It can be observed that both our method and AttUnet, which also includes attention modules, perform better than the other ones without attention modules. This proves that adding attention modules surely enhances the segmentation procedure by putting more attention into the tumor location. Oktay et al. have reported the same observation in their work with MSConvLSTM-Att [106].

Our architecture outperforms AttUNet with a significant margin, this is mainly due to the focus on location attention modules, besides the use of powerful feature extraction modules (ConvLSTM, SE-Inception and SE-ResNet) in the first part of the architecture, which is beneficial in eliminating irrelevant features. Our proposed architecture can be implicitly considered as a cascaded architecture even though we do not explicitly use multiple cascaded architectures.

Fig.3.7 displays a sample of the input MRI images, ground truth and the segmentation results using our proposed architecture. As seen in Table 3.1, the ET segmentation has the smallest DSC value. It can be seen also in Fig.3.7, where the ET region is not well detected especially in the first and third row.

It has to be mentioned that our method is slightly slower compared to the other methods, which is normal given the fact that complex building blocks has been used in order to ensure a better segmentation result.

### 3.5 Conclusion

In this chapter, we proposed a novel deep learning architecture for brain tumor segmentation we call multi-scale ConvLSTM Attention Neural Network, and we compare its performance to various deep learning architectures that are tailored to such kind of tasks. Our proposed method is built as a multiscale architecture composed of different state-of-the-art feature extraction blocks such as Inception, Squeeze-Excitation, Residual Network,
Fig. 3.7. Segmentation results sample: (a) is the input MRI images, (b) is the ground truth and (c) is the segmentation results using our proposed architecture.
ConvLSTM and finally Attention units. We compare the performance of our architecture to standard U-net, AttU-net and FCN that have shown effective results in semantic segmentation. Experimental results show that our proposed model outperforms standard U-net, AttU-net and FCN in terms of dice score. Our model reached 0.797 as a mean dice score for the three parts of the brain tumor, while Attention U-net, standard U-net and FCN reached 0.786, 0.736 and 0.728 respectively. We observe that both our method and the AttU-net perform better than the other ones, which can be explained that the integration of attention modules enhances the segmentation procedure. Besides, our method outperforms the AttU-net, and this is due to the use ConvLSTM, SE-Inception and SE-ResNet.
Conclusion

In this part, we used one of the most effective deep learning models for medical image segmentation, U-Net, to segment lung parenchyma in this section. This model can produce high-quality segmentation results with only a few hundred images in the data set. We achieved a dice coefficient index score of 0.9502 using U-Net, which is a high score for this kind of task. This approach is general and can be used for many different medical image segmentation tasks. This work was a preliminary study to explore how deep learning models can help with medical image segmentation and a first step to design new deep learning models for this domain.

Thereafter, we presented a new deep learning model for brain tumor segmentation. We introduced a new deep learning model for brain tumor segmentation that we called multi-scale ConvLSTM Attention Neural Network, and we evaluated its performance against various deep learning models that are designed for such tasks. Our model consists of a multiscale architecture that combines different feature extraction blocks such as Inception, Squeeze-Excitation, Residual Network, ConvLSTM and Attention units. We benchmark our model against standard U-net, AttU-net and FCN that have proven effective in semantic segmentation. Experimental results demonstrated that our model surpasses standard U-net, AttU-net and FCN in terms of dice score. Our model achieved 0.797 as a mean dice score for the three parts of the brain tumor, while Attention U-net, standard U-net and FCN achieved 0.786, 0.736 and 0.728 respectively. We notice that both our model and the AttU-net perform better than the others, which can be attributed to the use of attention modules that improves the segmentation process. Moreover, our model beats the AttU-net, and this is due to the use of ConvLSTM, SE-Inception and SE-ResNet.

As a first step of this part, we managed to test existing deep learning architectures for medical image segmentation. Then, we proposed a new architecture to serve the same purpose. We only compared our proposed architecture to others in the same level due to lack of computational power.
Part III

Medical Image Classification
Introduction

In the previous part, we investigated the role of features extraction methods in reaching great accuracy results in medical image segmentation. In this part, we are exploring the same roles but for medical image classification.

In computer vision, image classification is one of the key components for a better understanding of image content. There are numerous image processing methods in the literature that serve very well this purpose.

CNNs have proven to be very powerful when it comes to image classification. It all comes to the way CNNs are made, as mentioned in the state-of-the-art chapter 1.3. The power of CNNs resides on the feature extraction phase, that is why most of the state-of-the-art architectures achieved the best performances only by contributing to the feature extraction part of a CNN. In fact, starting from 2012, CNNs have known the raise of their era, with the introduction of a novel architecture named AlexNet [68], this architecture proposed by Kirzhevsky et al. has been successfully applied to various application domains. It consists of five convolution layers each one followed by a max-pooling layer then three fully connected layers. The success key presented by AlexNet’s authors is the use ReLu, multiple GPUs and overlapping pooling. Back then, it was very recurrent to use hyperbolic tangent (tanh) as an activation function in ANNs, however in AlexNet authors replaced tanh by ReLU and performed very well. Besides, the use of overlapping pooling layers which led to reducing the error rate by 0.5% and enabled the network to less overfitting as claimed by the authors. Nevertheless, AlexNet was trained on multiple parallel GPUs due to limitations on the computational power.

Another CNN architecture called ZFNet [167] was proposed in 2013 by Zeiler and Fergus, it is a variation of AlexNet. With their architecture, authors won the first place in the ImageNet challenge in 2013 by reaching 14.8% error rate. Their secret ingredient was the use of an advanced visualization technique that was proposed by Zeiler in 2011 [168], which consisted of visualizing intermediate layers of AlexNet, that enables for a better understanding of its mechanism in regards of the choice of different parameters. With the beforementioned technique, Zeiler and Fergus managed to build ZFNet with the same number
of layers as AlexNet but with very enhanced parameters, such as the use of $7 \times 7$ convolution filters with stride of 2 in convolutional layers instead of $11 \times 11$, which proved to preserve much more pixel information. Secondly, they replaced sparse connections used in some layers in AlexNet due to the split of the training across two GPUs, by dense connections in ZFNet. Finally, adopting the transfer learning as a supervised pretraining strategy to boost the training of the network.

After ZFNet, the research team of Google proposed a new CNN architecture named GoogLeNet [143] popular by the name of Inception. Inception network outperformed all the state-of-the-art models in image classification and image detection in the ImageNet challenge 2014, by reaching a significantly low error rate of 6.67%. Compared to previous networks, Inception has shown remarkable accuracy in regards of image classification, it is mainly due to the way Inception architecture is built, authors introduced a new convolutional block called inception, this block contains skip connections that are known to expand the architecture depth-wise and width-wise without compromising the computational cost and surely with a step-up regarding the network’s accuracy. The inception module comes in two versions, the first version is a naïve inception module that consists of a combination of $1 \times 1$, $3 \times 3$ and $5 \times 5$ convolution layers besides alternative pooling layers. However, this version encounters multiple computational cost issues and generates an inevitable number of outputs that surely led to increase the computation complexity. Thus, the use of a $1 \times 1$ convolution layer in parallel with the naïve inception to decrease the dimensionality depth-wise. This strategy, helped in building a very deep architecture without worrying about the computational cost, and showed very robust memory efficiency during training [143]. Many variants of the GoogLeNet architecture were released after, such as Inception V2 and V3 [144], Inception V4 [144] and Inception-ResNet [144]. Each one of them introduced an enhancement to the first version by including either convolutional or pooling blocks in an efficient way to decrease the computational cost and yet increase the accuracy. They surely showed better accuracy results compared to the original architecture.

The same year as the Inception network was released, another CNN architecture has drawn attention, it is called VGG [135]. It is a very popular CNN architecture and has a remarkable reputation inside the deep learning networks family. The VGG architecture is a variant of the AlexNet and ZFNet, instead of using $7 \times 7$ of $11 \times 11$ convolutional filters, authors of decided to use consecutive $3 \times 3$ convolutional filters. As stated in [135], the adoption of two consecutive $3 \times 3$ convolutional filters results in an effective receptive field of $5 \times 5$ convolution filter, and three consecutive $3 \times 3$ convolutional filters give a receptive field of $7 \times 7$ filters. The idea behind using consecutive $3 \times 3$ convolutional filters is to keep the same receptive field while using far less hyperparameters. Furthermore, experiments
showed that the decision function became more discriminative, and that is due to the raise of the number of non-linear rectification layers. The VGG architecture contains two different kinds of building blocks, the ones with two consecutive $3 \times 3$ convolutional filters and the other ones with three consecutive $3 \times 3$ convolutional layers, each followed by a max-pooling layer. The overall architecture contains two blocks of the first building block followed by three building blocks of the second kind. Finally, three fully connected layers added for discriminative purposes. The VGG architecture was adopted in many works in image processing, for image classification and image segmentation as we mentioned in each general introduction of each chapter from part II.

The VGG network was not the last contribution to the ImageNet challenge. In fact, ResNet [46] is another CNN based architecture that is known for its robust accuracy in image classification, object detection and image segmentation. It was introduced in 2015, in the ImageNet challenge and lowered the error rate of the GoogLeNet by 3.07% error rate. Its contribution resides in proposing a new building block for CNNs family called residual learning block, which consists of mostly $3 \times 3$ convolutional layers and pooling layers with stride of two. At the end of the network, a global average pooling is adopted followed by a fully connected layer with Softmax as a discriminative function. The residual learning module can be considered as a forward neural network with shortcut connections that help skipping one or more layers in the network. Despite of the high number of used layers in the network, ResNet architecture has no added extra parameters and no computational complexity, that is due the nature of skip connections between layers which performs an element-wise addition of the outputs of the identity mapping and the outputs of the stacked layers. Given that, ResNet can achieve low error rates and high accuracy.

Nonetheless, the ResNet team was not the one that reached the lowest error rate in the ImageNet challenge. In 2017, Jie et al. proposed an architecture called Squeeze-and-Excitation [61] and achieved 2.25% error rate. This architecture is composed of stacked multiple squeeze-excitation (SE) blocks. SE block contains a global average pooling operation instead of max pooling due to its superiority in terms of keeping global contextual features. SE blocks are considered the building blocks of this architecture, they are used in order to recalibrate features by aggregating feature maps across their spatial dimensions, this step is called squeeze and it generates embeddings of the global distribution from the channel-features. On the other hand, following up using excitation block to produce per-channel modulation weights. The contribution of this architecture resides on allowing global information features flowing through the network layers to selectively generate informative features while blocking useless ones.
Chapter 4

Feature Extraction Methods for Lung-Nodule Detection: A Comparative Deep Learning Study

Contents

4.1 Overview ................................................................. 65
4.2 Methods ................................................................. 66
   4.2.1 Restricted Boltzmann Machines ................................. 66
   4.2.2 2D Discrete Fourier Transform ................................. 68
   4.2.3 Principal Component Analysis ................................. 68
   4.2.4 Convolutional Neural Networks ................................. 69
4.3 Experimentation and results ........................................ 70
4.4 Conclusion .............................................................. 74

4.1 Overview

Feature extraction methods were one of the success keys that made a breakthrough in deep learning era, by providing relevant features for training.

To achieve a significant accuracy in visual recognition systems it is mandatory to adopt an effective feature extraction method. Feature extraction is the first prerequisite step to an efficient image recognition system, that is due to its ability in representing data in simplified
and summarized form which affects positively all its following steps. In our work here, we focus on adopting CNNs in the medical field; specifically in lung nodule detection.

Therefore, in this chapter we present an in-depth comparative study of four different feature extraction techniques adopted to analyze a lung nodule image data set then we report their result performance within a deep learning scenario.

Experimental results show that feature extraction using convolutional neural networks reach the best results among other methods including restricted Boltzmann machines.

The rest of this chapter is organized as follows: next section introduces feature extraction methods that are used for this research. Then in next section, we present and discuss experimental results from our study. Finally, concluding remarks.

4.2 Methods

Feature extraction methods for image processing are usually divided into two categories: Global feature extraction methods: consist of extracting features that characterize the image at hand as a whole. These kinds of features are usually used for object detection or classification. There are several examples of feature extraction methods that are considered global, from which we mention Principal Component Analysis PCA [56], Fourier transform [48] and RBMs [123].

Local feature extraction methods: consist of selecting features that characterize image patches which are computed at multiple points in an image. Local feature methods are commonly adopted in object recognition. Some examples of local feature methods are; scale invariant feature transform SIFT [91] adopted in [30] for palm print recognition, Gabor Wavelet [72] utilized in [130] to perform face recognition.

Our work belongs to the use of global feature extraction methods due the nature of the used images, which are used for lung nodule detection. Thus, in the upcoming sections, we present a full description of the used global feature extraction methods in our work.

4.2.1 Restricted Boltzmann Machines

RBMs are one of the key components that started the deep learning era, it is the building block of Deep Belief Networks DBN that adopted a new training strategy back then called greedy-layer-wise as stated in early sections. RBMs are based on Boltzmann Machines, which were introduced by Hinton and Sejnowski in 1986 [53]. BMs are considered as neural networks with stochastic units bidirectionally connected. It is an energy-based model trained
in an unsupervised way (the probability joint distribution in (4.1) is defined by the model variable using an energy function).

\[ P(x) = \frac{e^{-E(x)}}{z} \]  

(4.1)

\( E(x) \) is the energy function:

\[ E(x) = -x^T W x - b^T x \]  

(4.2)

Where \( W \) here denotes matrix of weights belonging to the model, \( b \) presents the offsets for each \( x \), then \( z \) is the partition function that ensures the sum of \( P(x) \) equals to 1.

\( BMs \) take as input a set of unlabeled binary victors. Their training consists of updating repeatedly the weights among the units until reaching a state of equilibrium.

In the beginning, \( RBMs \) were proposed as a concept by Smolensky [138] in 1986. \( RBMs \) are simply \( BMs \) that are constituted of only two layers (hidden and visible) with restriction to form connections between units of the same layers. This restriction is the key concept that makes \( RBMs \) quickly reach an equilibrium state in the hidden layer given a set of visible units (for example, when using \( RBMs \) for image reconstruction, the state of equilibrium is reached when the image is optimally reconstructed).

To train a \( RBM \), Gibbs sampling is adopted [138]. Gibbs sampling is a simple Markov Chain Monte Carlo \( MCMC \) algorithm that is responsible for producing sample from the joint probability distribution of multiple random variables. To elaborate more, performing Gibbs sampling while starting with random state in one layer we can generate relevant features from a \( RBM \). Gibbs sampling relies on two steps:

- Sample \( h(t) \sim P(h|v(t)) \), sample all the elements of \( h(t) \) given \( v(t) \).

- Sample \( v(t+1) \sim P(v|h(t)) \), sample all the elements of \( v(t+1) \) given \( h(t) \).

Where \( h(t) \) denotes the hidden units and \( v(t) \) denotes the visible units. The mentioned steps above are performed repeatedly until reaching the equilibrium state, in other words until convergence.

Note that there are two commonly used strategies for approximation, they are adopted in order to make the training process more significant, they are called contrastive divergence \( CD \) [50] and stochastic maximum likelihood \( SML \) [37].

The \( CD \) algorithm is much powerful when it comes to modeling high-dimensional data, thus it is widely used in shallow models. To perform with \( CD \) algorithm, it takes as input
the visible units, the training rate and the number of Gibbs steps. Then the outputs are
RBM weights matrix and the gradient approximations noted as $\Delta w_{ij}$, $\Delta a_i$ and $\Delta b_j$. The
algorithm consists of randomly initializing weights with positive values between 0 and 1,
then performing Gibbs sampling steps between hidden and visible units to finally calculate
the gradient approximations.

### 4.2.2 2D Discrete Fourier Transform

In different application domains, the Fourier Transform has shown its importance and
effectiveness, such as in mathematics, engineering and physics. Its discrete equivalent is
called the DFT which is calculated using the fast Fourier Transform FFT, usually used in
signal processing to get from a spatial representation its equivalent spectral representation.
In our work here, we mainly deal with image data which are considered a form of signals
represented in 2D, those signals are discretely sampled at constant intervals and are of finite
durations. In this kind of situations, DFT seems convenient due to the need of only finite
number of sinusoids.

The 2D-DFT $F$ and its inverse $f$ are mathematically represented by the following formu-
las:

$$F(u, v) = \frac{1}{NM} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)} \quad (4.3)$$

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F(u, v) e^{j2\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)} \quad (4.4)$$

where $0 \leq u \leq M-1, 0 \leq v \leq N-1, 0 \leq x \leq M-1$ and $0 \leq y \leq N-1$.

### 4.2.3 Principal Component Analysis

PCA is a mathematical function that resides on an orthogonal transformation in order
to convert a data of possibly correlated variables to a data of uncorrelated variables named
principal components. The effectiveness of PCA relies on reducing the dimensionality of
the high-dimensional data at hand by preserving the variance in its distribution as much as
possible.

If we assume that each data point is represented by a vector $x$ with $N$ dimensions, PCA
starts with finding a linear function $\alpha 1(x)$ with a maximum variance of the projection of data
points onto it. Then, PCA tries to look for another linear function $\alpha 2(x)$ with a maximum
variance of the projection of the data points onto it among all lines orthogonal to $\alpha 1(x)$.  

4.2 Methods

this process is repeated until a $\alpha k$ with the maximum variance for some $k > 1$ is found. To summarize, the main goal of PAC is to reduce as much as possible the dimensionality of a given high-dimensional data set by making most of the variation in the data set accounted for by the first $k$ principal components, then reducing the complexity obtained by transforming the variables.

4.2.4 Convolutional Neural Networks

CNNs are one of the most used architectures in a variety of machine learning tasks during the last few years, they have proven to be very effective in face recognition, object detection, text mining, action recognition, and many more. A standard CNN architecture consists of two different types of layers: convolution layers and pooling layers. Note that in this work, the fully connected layers are not included when CNN is only used for feature extraction.

Convolution layers: it simply consists of a NxN filter applied to input data to extract feature maps that contain high level features used as an input for next layers. In the literature, this kind of layers are followed by a nonlinear activation function in order to increase the expressiveness of the generated feature maps. Mathematically, a convolution operation can be represented by the following formula:

$$y_k = f(W_k \ast x) \quad (4.5)$$

where $x$ represents the input image, $W_k$ denotes the used convolution filter in relation to the $k^{th}$ feature map, and the convolution operation is represented by the multiplication sign. The nonlinear activation function mentioned above is represented by the $f(\cdot)$ function, in our case it is the rectifier linear activation ReLU function [99].

Pooling layers: this kind of layers have the ability to control overfitting in a network by progressively reducing the spatial size of feature maps using maximum or average operations on data, pooling layers are usually used after convolution layers to reduce the number of parameters and decrease computation capacity in a network. There are various type of pooling layers, Max-pooling and average-pooling are the most common ones in the literature. Max-pooling is represented by the following formula:

$$y_{kij} = \max_{(p,q)\in R} x_{kij} \quad (4.6)$$

Where $x_{kij}$ denotes the element at location $(p,q)$ contained by the pooling region $R_{ij}$. 
4.3 Experimentation and results

This section describes the experiments conducted to compare the performance of each feature extraction technique adopted in our comparative study. To conduct our experiments, we use the lung image database consortium image collection (LIDC-IDRI) [4]. It contains many diagnostic and lung cancer CT scans with annotated lesions. It is a web-accessible resource for the development of CAD methods for lung-cancer segmentation and diagnosis. The advantage of this data set is that it is provided with descriptive files that give details about any lung nodule location and size in the CT scans. In our study, we are interested in the lung nodule area more than other parts of the lung. Hence, we developed an algorithm that consists of extracting lung nodule patches in a $32 \times 32$ size, from the CT scans with the help of annotations provided with the data set. Thus, our data set contains image patches containing lung nodules (Fig. 4.1(a)) and images patches without lung nodules (Fig. 4.1(b)). For accuracy enhancement measures and to increase the volume of our data set, we applied data augmentation methods such as translation, rotation at different degrees, flipping and scaling. In total, our data set contains 8000 image patches, out of which 70% are for the training and 20% for validation then 10% designed for testing.

CNNs are known for their ability to retrieve complex feature characteristics, which make them very appealing for object detection tasks. In our work here, we compare a standard CNN constituted of the parts described before, to the other feature extraction methods mentioned in the same section. Fig. 4.2 presents the data processing flow used in our experiments. Once image patches are extracted, a data augmentation mechanism is applied to proceed with the
4.3 Experimentation and results

Fig. 4.2. Overview of the data processing flow (from data pre-processing to classification).

Thereafter, each feature extraction method is followed by a deep neural network DNN in order to measure the effectiveness of each feature extraction method.

The first feature extraction method consists of a standard CNN. The architecture is composed of a convolutional layer in after the input, that produces a set of feature maps with convolution filter bank composed of $32 \times 32$ filters. Next, we fed those feature maps to a ReLU activation function followed by a batch normalization layer [59]. Next, a max-pooling layer is applied to the output with $2 \times 2$ window with a stride of 2 pixels. We repeat those steps two more times but with different number of parameters, such as the number of convolution filters is 64 and 128 in the second and the third runs respectively.

The second feature extraction method we experimented with consists of an RBM. The beneficial effect of RBMs while applied for feature extraction is tremendous as proposed by Hinton [52], because they use hidden units to model the correlation among raw features
4.3 Experimentation and results

in the processed data. Our implemented RBM is composed of 1024 \((32 \times 32)\) visible units and 512 units in the hidden layer, then we trained our model with the CD learning algorithm using 30 Gibbs steps.

In our third method, we adopted 2D-DFT for feature extraction as used by Tao et al. [145]. We adopted Eq.4.3 in order to transform all images using 2D-DFT transformation, we then took the magnitude coefficient matrices for the classification step.

For the last method, we adopted PCA for feature extraction by reducing the dimension of the input data from 1024 into 512 components. After each feature extraction method, a DNN with the same parameters was introduced for classification purposes in order to measure their performance. The used DNN is composed of three hidden layers, each layer consists of 512, 1024 and 128 neurons in the first, the second and the third layer respectively. The model gives as output two options, the first is for a patch with a lung nodule and the second is for a patch without a lung nodule. All layers contain a ReLU method as activation function, except for the output where we use the sigmoid function in order to compute the probabilities of the outputs.

It is known in the literature that when initializing a model with the right parameters, the model has more chances to converge rapidly. Therefore, in our experiment we adopt the He initialization [45]. It consists of performing random initialization then multiplying the random parameters by for the links between layers \(l - 1\) and \(l\), where \(S(l - 1)\) stands for the size of layer \(l - 1\). As reported in [41], the main reason after using He initialization in our work is that it reportedly leads to faster training. Overfitting is one of the main cons of neural networks, to tackle this issue we adopted two different regularization methods. First, we performed L2-regularization [102] that consists of adding a regularization term to the error function so that it keep decreasing in the following form:

\[
L_T(W) = L_D(W) + \lambda \cdot L_W(W)
\]

Table 4.1. DNN accuracy for the four feature-extraction methods.

<table>
<thead>
<tr>
<th>Methods</th>
<th>CNN</th>
<th>2D-DFT</th>
<th>RBM</th>
<th>PCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training accuracy</td>
<td>0.996</td>
<td>0.990</td>
<td>0.780</td>
<td>0.608</td>
</tr>
<tr>
<td>Validation accuracy</td>
<td>0.973</td>
<td>0.965</td>
<td>0.730</td>
<td>0.580</td>
</tr>
<tr>
<td>Test accuracy</td>
<td>0.962</td>
<td>0.943</td>
<td>0.655</td>
<td>0.509</td>
</tr>
</tbody>
</table>

\(W\) here denotes neural networks weights, \(L_T(W)\) is the total loss and \(\lambda\) is regularization parameter. \(L_D(W)\) is a sum-of-squares error function between the desired output and the network output, and \(L_W(W)\) is the sum-of-squares of the weight parameters. The second
4.3 Experimentation and results

The regularization method adopted is the dropout [54]. This regularization method relies on dropping out most of the irrelevant units during training. In Table 4.1, we present model results in terms of accuracy for all the methods we experimented with. We notice that the CNNs as a feature extraction method surely leads to the best accuracy as expected: 0.996 in training, 0.973 in validation and finally 0.962 in test time. Then comes the 2D-DFT is the second place by reaching 0.990 during training, 0.965 in validation and 0.943 in test time. The third place belongs to RBM, which achieved 0.78 in training, 0.730 in validation and 0.655 in test time. The reason behind this low accuracy may be due to the nature of RBMs in being shallow models which makes them need huge amounts of data to achieve a better accuracy. Its power can manifest when it is used multiple times to form a deep belief network, which is beyond the scope of this work. Lastly, the PCA was the worst feature accuracy method in terms of accuracy. It reached 0.608 in training phase, 0.580 during validation and 0.509 in test time. We observe that PCA is the only linear feature-transformation method we experimented with, we can argue that non-linear feature transformation tends to lead to higher accuracy results given the presented experiments. Fig.4.3 provides the accuracy plots for all the used feature-extraction methods. It presents a more detailed illustration of the performance of the four feature-extraction methods and shows that CNN outperforms other three.

Table 4.2. DNN loss for CNN and 2D-DFT.

<table>
<thead>
<tr>
<th>Methods</th>
<th>CNN</th>
<th>2D-DFT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training loss</td>
<td>3.6%</td>
<td>8.4%</td>
</tr>
<tr>
<td>Test loss</td>
<td>4.1%</td>
<td>12.3%</td>
</tr>
</tbody>
</table>

We observe that the 2D-DFT is not significantly worse than CNN in terms of accuracy. But when we compare their variance and bias, we notice that the CNN shows low variance and bias compared to the 2D-DFT. Which means that the Fourier transformation misses relevant correlations among features and target outputs, we call that underfitting. Table 4.2 shows the loss during training and test time for the CNN and the 2D-DFT from which we can get their bias and variance. Given the presented results in Table 4.2, we observe that the 2D-DFT has high loss and high bias in both training and testing compared to CNN.

Given the nature of data we experimented with (lung nodule images) and due to the sensitivity of the result in affecting the life of a patient, the detection of a nodule before it transforms into a cancer is a very important task. Therefore, adopting a model with high accuracy and low bias and variance is a very important matter and can be helpful for doctors in taking critical decisions and to save lives.
Fig. 4.3. Accuracy charts for the four different feature extraction methods in validation and training.

4.4 Conclusion

In this chapter, we perform a comparative study of four different feature extraction methods based on deep neural networks for lung nodule detection. Our data follows a flow that begins with data preparation, then features are extracted using each method separately. Then, for evaluation purposes, we adopted a deep neural network composed of three hidden layers. Experimental results show that CNN outperforms RBM, PCA and 2D-DFT, with 2D-DFT being very close to CNN in terms of accuracy but suffers from both high bias and high variance which leads to overfitting and affects the inference phase. The high classification accuracy reached after feature extraction with CNN is an evidence that CNN is very successful in extracting high-level features from CT scans of lungs for the purpose of lung-nodule detection.
Chapter 5
Contributions to Mixed-Pooling layers

Contents

5.1 Introduction ................................................................. 75
5.2 Related work ................................................................. 76
5.3 Fully Mixed Max-Average Pooling for Convolutional Neural Networks 80
  5.3.1 Fully mixed max-average pooling ................................ 80
  5.3.2 Results and discussion .............................................. 81
5.4 Mixed Pooling-Dropout for Convolutional Neural Networks Regularization 84
  5.4.1 Mixed pooling dropout .............................................. 84
  5.4.2 Experimental results ............................................... 86
5.5 Conclusion ................................................................. 92

5.1 Introduction

Pooling layers play an interesting role in reaching higher accuracies using CNNs. They are responsible for the down-sampling operation that aims to prevent overfitting in CNNs.

There are various conventional pooling operations that work fine for that purpose, such as max and average pooling which are the most popular ones in the literature. However, there are some variants that have proven to be more efficient in terms of preventing overfitting; those variants are a mixture of features generated from max and average pooling and combined in an optimal way. In this regard, we proposed a new mixing strategy between max and average pooling, called fully mixed-max-average pooling FMMAP, which outperformed max and
average pooling in terms of accuracy and significantly outperformed other mixed-pooling methods time-wise. Furthermore, the dropout showed remarkable results, it has the benefit of using a combination of multiple networks in one architecture besides preventing the network’s units from co-adapting in an excessive way. However, the application of dropout function is only limited to fully connected layers. To further address the problem of overfitting, we present a novel method called Mixed-Pooling-Dropout which benefits from the utilization of the dropout strategy in early stages of CNNs, especially in pooling layers. We represent the dropout function by a binary mask with each element drawn independently from a Bernoulli distribution.

We employ those architectures in the medical field, the first one is used to perform lung nodule classification, where we managed to outperform max and average pooling in terms of accuracy, and while being close to other mixed-pooling methods accuracy-wise, we significantly outperformed them time-wise. In addition, the second architecture is applied to brain tumor classification, it outperformed conventional pooling methods as well as the max-pooling-dropout method. With our method, we reached 0.926 vs 0.868 regardless of the retaining probability.

In the upcoming sections, we present thoroughly each one of the architectures, besides the experimental results.

5.2 Related work

Pooling layers have played a game changing role in the raise of CNNs in computer vision, due to their structure that helps in reducing the number of features, surprisingly, without affecting those that seem to be relevant for the model training. As a matter of fact, decreasing the number of features in this way have a direct beneficial effect on the performance of a CNN; for example, it helps in preventing overfitting and achieving better results in less time. In the deep learning literature, the well-known pooling functions are max, average and stochastic pooling. However, many variants of these functions have been proposed as well, such as mixed [162], mixed max-average [71], gated pooling (GP) function [71] and max-pooling-dropout [155]. Herein, we present each of these functions. In general, any pooling function can be represented by the following formula in Eq.(5.1).

$$a_{j}^{l+1} = \text{Pool}(a_{1}^{(l)}, \ldots, a_{i}^{(l)}, \ldots, a_{n}^{(l)}), a_{i} \in R_{j}^{(l)}$$

(5.1)

In Eq.(5.1), $R_{j}^{(l)}$ is the $j^{th}$ pooled region at layer $l$. The pooled region slides over a layer $l$ by a stride $s$, and $n$ is the number of elements in the pooled region. Pool() denotes the
5.2 Related work

pooling function performed over the pooled region.

Fig. 5.1 illustrates how a pooling function works in general. The input represents layer \( l \) and the red region is pooled region \( R_j^{(l)} \) where \( j \) in this case is 1 and the output \( X \) is \( a_j^{l+1} \) in the formula and the stride \( s \) is 2 in this example.

![Illustration of a pooling operation.](image)

The pooling operation can be either max or average. Both pooling operations have been used in many works, and have proven very effective in boosting the accuracy [98, 83]. The max pooling operation takes the maximum value in the pooled region as shown in Eq. 5.2, while the average pooling operation takes the average value over the pooled region according to Eq. 5.3.

\[
Max(x_{k_{ij}}) = \max_{(p,q)\in R_{ij}} (x_{k_{ij}}) \tag{5.2}
\]

\[
Avg(x_{k_{ij}}) = \frac{1}{|R_{ij}|} \sum_{(p,q)\in R_{ij}} x_{k_{ij}} \tag{5.3}
\]

In both Eq. 5.2 and Eq. 5.3, \( x_{k_{ij}} \) represents the element at location \( (p,q) \) covered by the pooling region \( R_{ij} \). Zeiler et al. [166] proposed a new probabilistic function for the pooling operation, called stochastic pooling. With stochastic pooling, the selected output is drawn from a multinomial distribution formed by units within a local pooled patch. A multinomial distribution consists in computing the probabilities \( p \) for each region (see Eq. 5.4) and then the activation \( a_j \) is sampled from the multinomial distribution based on \( p \) (see Eq. 5.5).
\[ p_i = \frac{a_i}{\sum_{k \in R_j} a_k} \quad (5.4) \]

\[ s_j = a_l \text{ where } l \sim P(p_1 \ldots p_k) \quad (5.5) \]

Even though these functions can work very well on many data sets, they still encounter some problems and it is unknown which one can be best for solving new problems. It is proven theoretically [11] and empirically [162] that the choice between the pooling functions is dependent on the characteristics that are present in the data. Therefore, as a solution, Yu et al. proposed a new method called mixed-pooling (MP) [162] that combines the two popular pooling functions max and average. The idea behind it is to randomly choose either of the two pooling methods (average or max) according to Eq.5.6.

\[ y_{kij} = \lambda \cdot \max_{(p,q) \in R_{ij}} x_{kij} + (1 - \lambda) \cdot \frac{1}{|R_{ij}|} \sum_{(p,q) \in R_{ij}} x_{kij} \quad (5.6) \]

Where \( x_{kij} \) represents the element at location \((p,q)\) covered by the pooling region \( R_{ij} \) and \( \lambda \) can be either 0, i.e. average pooling operation is performed, or 1, i.e. max pooling is performed. Thus, the pooling regulation scheme becomes a probabilistic matter which helps in overcoming the aforementioned issues. Furthermore, Lee et al. proposed three new methods called mixed max-average, gated and tree pooling [71]. The first method, mixed max-average pooling (MMAP), combines max and average pooling proportionally, in other words the output is a combination of fixed proportions extracted from max and average pooling using Eq.5.6. In this case \( \lambda \in [0,1] \) is a scalar representing the mixing proportion which specifies the exact amount of combination of max and average pooling. The difference between MP and MMAP is that the output of the former is the result of max or average pooling given that \( \lambda \) is either 1 or 0, while the output of the latter is a combination of the max and average operations given that \( \lambda \) can take any value between 0 and 1. In other words, MMAP is a generalization of MP. MMAP has the drawback of being “non-responsive”, i.e. the mixing proportion remains fixed regardless of what characteristics are present in the pooled region. The second method proposed by Lee et al., i.e. gated pooling (GP), addresses this nonresponsive behavior. Rather than fixing the mixing proportion, a gating-mask is learned that has the same spatial dimensions as the pooled region. Then the inner product of the gating-mask and the region being pooled produces a scalar which is fed through a
sigmoid function to generate the mixing proportion (see Eq.(5.7)).

\[
y_{kij} = \sigma(\omega^T R_{ij}) \cdot \max_{(p,q) \in R_{ij}} x_{kij} + (1 - \sigma(\omega^T R_{ij})). \frac{1}{|R_{ij}|} \sum_{(p,q) \in R_{ij}} x_{kij}
\]

(5.7)

In (5.7), \(x_{kij}\) represents the element at location \((p,q)\) covered by the pooling region \(R_{ij}\); \(\omega\) denotes the values of the gated mask and \(\sigma\) is the sigmoid function in Eq.(5.8).

\[
\sigma(z) = \frac{1}{1 + \exp^{-z}}
\]

(5.8)

The third method proposed by Lee et al. [71] has three main points: first it learns pooling filters directly from data. Second, it allows to learn how to mix pooling filters in a differentiable way, thus the whole function will be differentiable with respect to its parameters and inputs. Finally, the tree pooling brings together the other characteristics belonging to the structure of a hierarchical tree. Each leaf node is associated with pooling filters learned during training denoted as \(v_m\), with \(m\) from 1 to \(n\) (number of leaves). The internal nodes in turn are processed in the same way as the gated max-average pooling method. Finally, the root node corresponds to the overall output produced by the tree (see Eq.5.9). For a specific example of a “2 level” tree pooling, the resulting function will be as follows:

\[
f_{\text{tree}}(x) = \sigma(\omega_1^T x)v_1^T x + (1 - \sigma(\omega_2^T x))v_2^T x
\]

(5.9)

\(x\) represents the input and \(\omega\) is the gating mask similar to the one adopted in gated pooling.

Moreover, max-pooling-dropout is another pooling related method that combines max pooling and a Dropout function. Dropout is a regularization approach that is very efficient in overcoming overfitting in artificial neural networks and deep neural networks. In [54], N. Srivastava et. al. published a paper describing in details how does a dropout function work. It acts in fully connected layers by randomly dropping a number of hidden units given a probability \(p\) named retaining probability. This approach has been adopted in several works, most of them are winners of the ImageNet competition, such as AlexNet [68], SENet [61], Inception [143] and many more. As shown in [155], the dropout function was extended to cover feature extraction layers using a method called max-pooling-dropout. It consists of selecting activations based on a multinomial distribution then perform max-pooling operation, in this way the output is stochastically picked instead of just selecting the strongest activation.
5.3 Fully Mixed Max-Average Pooling for Convolutional Neural Networks

Another dropout inspired work called dropconnect was introduced in [147], it consists of randomly selecting a set of weights to set to zero instead of setting randomly picked units to zero, their motivation behind using this method is to address the shortcoming presented by the dropout function which consists of preventing network weights from collaborating with one another in order to remember training samples. In [38], authors proposed a novel approach related to dropout called maxout network. The maxout function improves both optimization and model averaging characteristics in a dropout function. Adopting a function like dropout in pooling layers is a great way to enhance the quality of selected features in order to reach a better learning scheme.

5.3 Fully Mixed Max-Average Pooling for Convolutional Neural Networks

5.3.1 Fully mixed max-average pooling

There is one noticeable issue with the aforementioned mixed pooling operations, they randomly choose between one of the standard pooling operations or combine them proportionally. Which raises an interesting question, what if fully combined max and average pooling in one single layer? Before answering this question, we are going to describe our architecture in details. The standard pooling layer is replaced by a layer that combines max and average pooling operations, in order to provide different types of features such as global contextual feature information provided by average pooling and dense features provided by max pooling. The proposed mixed pooling block is illustrated in 5.2. Mathematically speaking, our proposed method can be represented by the following formula:

![Fig. 5.2. Fully mixed max-average pooling block.](image-url)
5.3 Fully Mixed Max-Average Pooling for Convolutional Neural Networks

\[ y_{kij} = f(\max_{(p,q)\in R_{ij}} x_{kij}) \oplus f\left(\frac{1}{R_{ij}} \sum_{(p,q)\in R_{ij}} x_{kij}\right) \]  

(5.10)

In Eq.(5.10), \( x_{kij} \) represents the element at location \((p, q)\) covered by the pooling region \( R_{ij} \); \( f \) represents the \(1 \times 1\) convolution filters and we use \( \oplus \) to denote the depthwise concatenation of the two operations’ outputs.

The use of both operations in one layer can surely slow the network, that is why we included a \(1 \times 1\) convolutional layer after each pooling operation before we combined them. The use of \(1 \times 1\) convolutional layer have a beneficial effect of reducing the outputs dimension and decreasing the computational cost [143]. The \(1 \times 1\) convolution was first introduced by Lin [83] for the purpose of cross channel down sampling. In other words, the utilization of \(1 \times 1\) convolution has the advantage of squeezing the information in a certain volume by reducing the number of channels.

5.3.2 Results and discussion

This section presents the results of our comparative study of different pooling-based architectures. The CNN architectures we experimented with are: using max-pooling, using average-pooling, using mixed-pooling, using mixed-max-average-pooling, using gated-pooling and finally using our proposed pooling method.

Fig. 5.3. The convolutional neural network architecture adopted for all the used models. Note that the pooling layer is replaced by the convenient pooling operation regarding each one of the six models.

In order to be neutral as much as we can in our comparative experiment, we used the same architecture with the same setup for all pooling methods. Fig.5.3, shows the used architecture for our experiment. It contains a convolution layer with 32 \(3 \times 3\) filters then followed by an activation layer with ReLU function and then a pooling layer. This block is repeated three times with 64 \(3 \times 3\) and 128 \(3 \times 3\) filters in convolution layers respectively. Then come three fully connected (FC) layers for the discriminative part of CNN. The FC layers contain 1024, 512 and 128 hidden units, respectively.
5.3 Fully Mixed Max-Average Pooling for Convolutional Neural Networks

We performed our experiments with the lung image database consortium image collection (LIDC-IDRI) [4]. It contains CT scans of lung-cancer screening with described and annotated lesions. It is an open-source data set, mainly dedicated for the development of CAD systems for lung-cancer segmentation and diagnosis. The advantage of this data set is that it is provided with descriptive files that give details about any lung nodule location and size in the CT scans. In this work, we are interested only in the lung nodule regions, thus we developed an algorithm that consists of extracting lung-nodule patches (32 × 32 pixels in size) from the CT scans using the annotations provided in the descriptive files. Hence, our data set has image patches that contain lung nodules (see Fig.5.4(a)) and image patches that do not contain lung nodules (see Fig.5.4(b)). We applied data augmentation methods such as translation, rotation at different degrees, flipping and scaling to increase the volume of our data set. In total, our data set consists of 8,000 examples out of which 70% are used for training, 20% for validation and 10% for testing.

Our proposed method was put under two types of experiments, the first one consists of comparing it with standard pooling operations (max and average). Then, our proposed method is compared to mixed-pooling-based methods. Table 5.1 shows accuracy results for our proposed method with max and average pooling. Our method outperformed both max and average pooling in terms of accuracy. We achieved 0.954 in training and 0.906 in test time, while max and average pooling achieved 0.947 in training and 0.901 in testing and 0.941 in training and 0.904 in test time respectively. As we predicted, the performance of max and average pooling is slightly the same. Nonetheless, in Table 5.1 it shows also the running time in seconds of the three pooling methods. Our model
Table 5.1. Model accuracy and time performance for max, average and our proposed pooling method.

<table>
<thead>
<tr>
<th>Networks</th>
<th>Average-pooling</th>
<th>Max-pooling</th>
<th>FMMAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training accuracy</td>
<td>0.931</td>
<td>0.937</td>
<td>0.954</td>
</tr>
<tr>
<td>Test accuracy</td>
<td>0.903</td>
<td>0.901</td>
<td>0.906</td>
</tr>
<tr>
<td>Performance Time</td>
<td>190.466</td>
<td>190.551</td>
<td>209.753</td>
</tr>
</tbody>
</table>

took faintly more time compared to max and average pooling, which is expected given the performing of both methods in the same layer.

Table 5.2. Model accuracy and time performance for all mixed-pooling architectures including our proposed method.

<table>
<thead>
<tr>
<th>Networks</th>
<th>FMMAP</th>
<th>MP</th>
<th>MMAP</th>
<th>GP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training accuracy</td>
<td>0.954</td>
<td>0.962</td>
<td>0.979</td>
<td>0.973</td>
</tr>
<tr>
<td>Test accuracy</td>
<td>0.906</td>
<td>0.922</td>
<td>0.938</td>
<td>0.956</td>
</tr>
<tr>
<td>Performance time</td>
<td>209.753</td>
<td>353.617</td>
<td>430.154</td>
<td>561.488</td>
</tr>
</tbody>
</table>

The second part of our comparative study is presented in Table 5.2. As it can be observed, all the four methods have relatively good accuracy results, with GP reaching 0.973 in training and 0.956 in test, MMAP with 0.979 in training and 0.938 in test, MP with 0.962 in training and 0.922 in test and our FMMAP with 0.954 and 0.906 in training and test, respectively. While GP has the highest accuracy on the test data set, the results are very close to each other for all the four models which makes it hard to pick a winner and gives evidence that a mixture of features coming from max and average operations can lead to some good results. Table 5.2 also shows the running times (in seconds) for the four models. FMMAP is significantly faster compared to the other three methods and this is mainly due to the $1 \times 1$ filters added after each pooling operation. In our proposed pooling block, performing both max and average pooling has surely increased the number of channels, hence the increase of number of parameters in the following convolution layer. Nevertheless, with the use of $1 \times 1$ convolution filters after each pooling operation and before jumping to the next convolution layer, we managed to reduce the number of channels by dividing it by two, therefore, decreasing the model’s number of parameters. The gated pooling model takes more time compared to the other models and this is essentially due to the use of sigmoid function in the pooling block.
5.4 Mixed Pooling-Dropout for Convolutional Neural Networks Regularization

5.4.1 Mixed pooling dropout

Mixed-Pooling-dropout is a fusion of a mixed-pooling layer with a dropout layer. The mixed-pooling layer consists of a mixture of both max and average pooling operations in a particular way.

Translation invariance is one of the problems faced while performing image processing, the utilization of pooling operations deals with such kind of problems efficiently. Pooling operations have also the beneficial effect of reducing the computational cost of a CNN model by canceling irrelevant features that are present in a feature map. Nevertheless, adding the dropout function to fully-mixed-pooling layers leads to significant improvements result-wise, and that is due to the dropout’s stochasticity and diversity of features. In this work, the pooling operation adopted is fully-mixed-pooling which is mathematically represented by the following formula:

\[ x_k \] represents the elements of the pooled region, \( f \) is the \( 1 \times 1 \) convolution operation performed over the pooling operation in order to computationally help the network and \( \oplus \) denotes the depth-wise concatenation of both operations’ outputs. Furthermore, given the output units’ nature in being binary with respect to a retaining probability, the adopted dropout function is represented by a binary mask containing elements independently drawn respecting a Bernoulli distribution. In [155], authors claim that applying pooling operation after performing a dropout is considered as if activation units were sampled from a multinomial distribution. We describe a multinomial distribution as follows; we first compute the probabilities \( p_i \) for each area following this formula:

\[
p_i = \frac{a_i}{\sum_{k \in R_j} a_k}
\] (5.11)

then based on \( p \), the activation \( a_l \) is sampled regarding a multinomial distribution:

\[
s_j = a_l \text{ where } l \sim P(p_1...p_k)
\] (5.12)

a dropout function can be included into feature extraction layers in a CNN in two ways: first, we can apply dropout after generating outputs from fully mixed-pooling layer. Or, we can perform dropout function on each pooling operation then mix both the outputs (max and average). In both scenarios, the dropout function can be represented mathematically respecting the following equation:
5.4 Mixed Pooling-Dropout for Convolutional Neural Networks Regularization

\[ z_k^{(l)} \approx M \ast y_k^{(l)} \] (5.13)

\( M \) represents a binary mask generated from a Bernoulli distribution with the same dimensions as \( y_k^{(l)} \), and \( \ast \) denotes an element-wise multiplication of the generated binary mask and the pooled area. \( y_k^{(l)} \) in Eq.?? takes two different values, depending on if the dropout function is performed after fully-mixed pooling then the resulted value from Eq.?? is taken, otherwise \( y_k^{(l)} \) take the resulted value from Eq.5.3 in case of average polling or from Eq.5.2 in case of max pooling. Performing this strategy grates that not only the maximum or the average value is being selected but the chance for other values that might be relevant to the training process is a valid option, this is possible regarding a retaining probability \( p = 1 - q \), where \( q \) here stands for the probability of a value to be drooped out. When a dropout function is used in fully connected layers during training time, at test time a mean network [140] is adopted to help reduce the error. A mean network contains all the hidden units but with their outgoing weights halved. In [155], authors used what they called ‘probabilistic weighted pooling’ during test time, which consists of averaging all the possibly trained max-pooling-dropout networks. In our case, we perform a multiplication of the output from the pooling layer with the retaining probability \( p \) as indicated by these following equations:

\[ d_k^{(l)} = p \cdot \text{Max}(x_k) \oplus p \cdot \text{Avg}(x_k) \] (5.14)

\[ d_k^{(l)} = p \cdot y_k^{(l)} \] (5.15)

With Eq.5.14 is the formula for performing dropout before mixing pooling outputs (\( \text{Max}(x_k) \) from Eq.5.2, \( \text{Avg}(x_k) \) from Eq.5.3) and Eq.5.15 is the formula for dropout after mixing max and average pooling features(\( y_k^{(l)} \) from Eq.??).

With Eq.5.14 is the formula for performing dropout before mixing pooling outputs (\( \text{Max}(x_k) \) from Eq.5.2, \( \text{Avg}(x_k) \) from Eq.5.3) and Eq.5.15 is the formula for dropout after mixing max and average pooling features(\( y_k^{(l)} \) from Eq.5.10). This strategy is equivalent to using a mean network scheme as adopted in fully connected layers. It is an efficient approach as we will be presenting in upcoming sections. In the next section, we provide empirical proofs that our proposed method is significantly superior to conventional pooling and max-pooling-dropout methods, but first we provide theoretical evidences to why our method outperforms those methods. The more included features from different sources (max and average) the merrier [162], to include more features is certainly good for CNN’s accuracy. Although, some irrelevant features might be taken into consideration while adopting this strategy, which has negative effect on model’s accuracy. Thus, excluding them is a mandatory
task. That is why, in our proposed method we perform a dropout function to stochastically remove any irrelevant feature units. This approach enhances the quality of our model’s generated features stochastically, and considered as if they were drawn from a multinomial distribution.

5.4.2 Experimental results

Our experiments were conducted using a data set of MRI images for brain tumors. The obtained data identified 120 patients at first, then ten patients were excluded because they did not have genomic cluster information available. Therefore, the final version of the data-set contains 110 patients gathered from five different institutions [92]. There were 101 patients with all available sequences, nine patients with a missing post-contrast sequence. The number of slices ranged from 20 to 88 among different patients. To determine the original growth pattern of the tumor, they analyzed only the preoperative data. Evaluation of the tumor shape was based on Fluid Attenuated Reversal Recovery (FLAIR) abnormality, because the tumor rarely improves in Low-Grade Gliomas (LGG). The FLAIR images were manually annotated by drawing the FLAIR abnormality on each slice to form training data for the automated segmentation algorithm. For this purpose, a locally developed software was used. Then a board eligible radiologist has reviewed all annotations and modified those that were incorrectly annotated. All image data with corresponding segmentation masks for each used case is publicly available via the following link: https://www.kaggle.com/mateuszbuda/lggmri-segmentation/version/2. Since we are performing medical image classification, we only use image data with their corresponding labels. To conduct our experiments, we needed a computational power that can satisfy our needs, we trained our model on NVIDIA TESLA P100 GPUs. CNNs are known to be very successful in computer vision in general, and in classification specifically, that is due to their ability to process massive amount of data. The building blocks of this architecture such as convolution and pooling are one of the keys to a significant accuracy, mostly used for retrieving complex and relevant features as demonstrated in [137]. In our work, we inspect the role of pooling layers in enhancing the accuracy of CNNs besides the use of the dropout function in feature extraction layers. For that, we introduce a novel pooling method named fully-mixed-pooling-dropout and we compare it first to conventional pooling methods then to max-pooling-dropout [155]. Our proposed method is presented in two versions, first version consists of max and average pooling, each one is followed by a $1 \times 1$ convolution for computational enhancement purposes, then both outputs are concatenated depth-wise and finally followed by a dropout layer, see Fig. 5.5(a). on the other hand, the second version of our proposed method contains a dropout layer applied to input feature maps, the output then is fed to max-pooling and average pooling.
separately and each one of them is followed by a $1 \times 1$ convolution layer, after that, the results are concatenated depth-wise as shown in Fig.5.5(b).

The whole architecture is presented as follows: a convolution layer of 64 filters with $3 \times 3$ shape and ReLU activation function, followed by a batch normalization layer then our proposed mixed-pooling-dropout layer (First model) with $2 \times 2$ window and stride 2 in both max and average pooling, Fig.5.5(a). The second architecture has the same parameters configuration except for the pooling area where we use the second version of our proposed method, Fig.5.5(b). Moreover, Fig.5.5(c) presents the max-pooling-dropout method. It consists of the same number of layers used in all the other architectures except for the pooling layers, max-pooling-dropout is used instead.

To train our networks we used 50 epochs with 32 as batch-size, and we used Adam as an optimization algorithm.

5.4.2.1 Mixed pooling dropout vs Max, Average and Stochastic pooling

Our method is a pooling-based function, it is mandatory to compare its performance with conventional pooling methods like max pooling, average pooling and stochastic pooling. Stochastic pooling is included in our experiments because it resembles to our proposed method in stochastically selecting activation units and showed promising results as shown in [151, 171]. This section contains a comparative experiment in which we compare the performance of conventional pooling methods with ours. We kept the same architecture description as described above (same number of layers, hyperparameters) in our method in the architectures of these conventional pooling architectures. The difference resides only in the used pooling method for down-sampling (max, average or stochastic). As presented in Table 5.3, our proposed method in its both versions overtake all of max, average and stochastic pooling accuracy-wise and also in sensitivity and specificity. As was assumed, max and average pooling achieved the same results with a slight difference, which is normal given that in the literature they both have always shown equal performance. Thus, the choice between them remains not very important. On the other hand, stochastic pooling showed slightly better performance compared to max and average pooling and that is due to its nature in stochastically drawing activation features.

5.4.2.2 Mixed pooling dropout vs Max pooling dropout

As mentioned above, one of the few methods that adopted dropout operation in feature extraction layers is max-pooling-dropout [155]. hence, it is mandatory to provide experimental results to compare its performance to our proposed method. Fig.5.6 shows accuracy and loss
Fig. 5.5. The three architectures adopted for the experimentation. Mixed-pooling-dropout first version (a), second version (b) and max-pooling-dropout (c)
Table 5.3. Proposed method’s accuracy, sensitivity and specificity compared to those of Max, Average and stochastic pooling.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Sensitivity</th>
<th>Specificity</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>0.843</td>
<td>0.849</td>
<td>0.850</td>
</tr>
<tr>
<td>Avg</td>
<td>0.850</td>
<td>0.844</td>
<td>0.855</td>
</tr>
<tr>
<td>Stochastic</td>
<td>0.858</td>
<td>0.854</td>
<td>0.864</td>
</tr>
<tr>
<td>Ours(mix first)</td>
<td>0.884</td>
<td>0.874</td>
<td>0.888</td>
</tr>
<tr>
<td>Ours(drop first)</td>
<td>0.918</td>
<td>0.920</td>
<td>0.926</td>
</tr>
</tbody>
</table>

Fig. 5.6. Model accuracy and loss charts for our proposed dropout-pooling method (v1 and v2) and max-pooling-dropout.

charts belonging to both versions of our proposed method opposing to max-pooling-dropout. We managed to show a better accuracy using our proposed method in its both versions compared to max-pooling-dropout, with the second version reaching the highest accuracy and the lowest loss. Furthermore, Table 5.4 presents model accuracy results for all the three methods. It can be observed that both models of our proposed method outperformed max-pooling-dropout with a significant margin, with second model of MiPD reached 0.926 in training and 0.908 in test, first model of MiPD with 0.888 in training and 0.864 in test, and finally max-pooling-dropout with 0.868 in training and 0.831 in test. In addition to accuracy, our proposed method reached better sensitivity and specificity as shown in Table 5.4.

Furthermore, Table 5.4 also shows the loss for the three methods, and again our proposed method in both models achieved a lower loss value compared to max-pooling-dropout with 0.274, 0.338 and 0.369 respectively. As shown empirically, the second model of our proposed method outperforms the first model, even with the same number of layers and parameters in both models. The only difference is that the dropout in the second model is performed twice before we include max and average pooling operations. While in the first model, we only
perform the dropout function one time and that is after the concatenation of both average and max pooling layers. The difference in the results between both models of our proposed method is explained by the fact that performing dropout twice in the second model, the randomness of selecting diverse and significant features is higher than when we performed the dropout function one time on the generated pooling features in the first model.

Table 5.4. Models’ accuracy, loss and time performance in seconds for our proposed method (both versions) vs max-pooling-dropout.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MaPD</th>
<th>Ours(mix first)</th>
<th>Ours(drop first)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training accuracy</td>
<td>0.868</td>
<td>0.888</td>
<td>0.926</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.858</td>
<td>0.884</td>
<td>0.918</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.861</td>
<td>0.874</td>
<td>0.920</td>
</tr>
<tr>
<td>Training Loss</td>
<td>0.369</td>
<td>0.338</td>
<td>0.274</td>
</tr>
<tr>
<td>Test accuracy</td>
<td>0.831</td>
<td>0.864</td>
<td>0.908</td>
</tr>
<tr>
<td>Performance Time</td>
<td>810.521</td>
<td>853.940</td>
<td>846.687</td>
</tr>
</tbody>
</table>

In addition to accuracy, sensitivity and specificity, time performance is also presented in Table 5.4. As we presumed, our model took slightly more time than max-pooling dropout and that is explained by the fact that a mixture of both max and average pooling operations is performed in the same layer. Nevertheless, utilizing dropout before/after pooling functions served greatly in reducing time performance simply by randomly disabling useless units for the training of our model. Besides, $1 \times 1$ convolution filters were added right after each combination of max and average pooling in order to decrease the number of generated feature maps for the same purpose which is to reduce time performance. During test time, we adopted the “mean-network” strategy and that means the test time network contains all units but with their outgoing weights divided by two. According to [155], this strategy is far more effective than taking the average predictions of several separate networks. Furthermore, we make certain that it is the same case in our experiments, empirical evidence shows that using mean-network at test time outperforms the averaging strategy, see Table 5.5.

Table 5.5. Mean-network vs. averaging strategy at test time for our proposed method.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Ours(mix first)</th>
<th>Ours(drop first)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean-network accuracy</td>
<td>0.864</td>
<td>0.908</td>
</tr>
<tr>
<td>Averaging accuracy</td>
<td>0.858</td>
<td>0.897</td>
</tr>
</tbody>
</table>

Our proposed method showed great performance in medical images. Unfortunately, its generalization to other application domains is out of the scoop of this study. Nevertheless,
5.4 Mixed Pooling-Dropout for Convolutional Neural Networks Regularization

Fig. 5.7. Training accuracy by retaining probability for our proposed mixed-pooling-droput method (v1 and v2) and max-pooling-droput.

Our proposed pooling block is composed of state-of-the-art blocks, which have shown very efficient results in many application domains [83, 141] (including medical domain [154]), their combination in a certain manner already proved to be effective for different application fields [26]. Given that, and until we empirically prove those assumptions in future works, we can presume that our proposed method can be generalized to other applications.

5.4.2.3 Dropout rate

The dropout function works with hyperparameter that is directly responsible for the activation units that will be dropped out of training, named retaining probability $p$. In [140], it is proven that $p$ makes various neural networks very efficient. $p$ can take values from 0 to 1. $p = 1$ means that no units were excluded from training while small values of $p$ mean that more units were disabled during training. In case of our proposed method, $p = 1$ denotes that mixed-pooling is performed while in max-pooling-droput $p = 1$ means that max-pooling is adopted. As shown in section ?? of chapter 5, we show that mixed-pooling layers reach promising results compared to conventional pooling methods. Including dropout function in fully connected layers has shown very good results, especially when choosing optimal values for the retaining probability $p$ regarding each type of layer. Different values for hidden layers are usually between 0.5 and 0.8, while the optimal value by experiments is 0.5. For input layers, the choice of $p$ is related to what kind of data is dealt with. According to [140], the optimal value that $p$ can take in input layers when processing image data sets is 0.8.
5.5 Conclusion

In the above experiments, we presumed that the optimal value for $p$ is 0.5 since it is the optimal value while using dropout function in fully connected layers according to [140]. However, to validate our presumption in regards to the optimal value for the retaining probability, we conducted more experiments where we try different value of $p$ for all the architectures. Fig.5.7 shows the impact of changing the retaining probability on the accuracy of each method. The second model of our proposed method outperforms the other methods in all scenarios where the model with the value $p = 0.5$ reached the highest accuracy of them all. Our proposed method (both versions) performs poorly when using higher values of $p$, while for smaller values it performs relatively better. As was presumed, the optimal retaining probability value remains $p = 0.5$. on the other hand, for max-pooling-dropout it the opposite, with $p = 0.5$ remains the optimal value in this case too. In general, regardless of what value take $p$, our proposed method outperforms max-pooling-dropout anyway.

5.5 Conclusion

In this chapter, we provided two different mixed pooling strategies. On the one hand, we presented a pooling-based comparative study with the proposition of a new mixed-pooling method, called fully mixed max average pooling. With our method, we perform a fully mixture of both max and average pooling operations in one single pooling layer with a $1 \times 1$ convolutional layer to reduce both the number of parameters and the computational cost. In this work, we trained our models on the lung image database consortium image collection LIDC-IDRI, for lung nodule detection. Experimental results show that our proposed method outperforms standard pooling methods accuracy-wise, while being 0.05% low on accuracy compared to state-of-the-art gated pooling method, with the leading best accuracy in this experiment. Nevertheless, our pooling method is 2.68 times faster than any of the mixed pooling strategies used in this chapter. To summarize, FMMAP provides a mixture of max and average pooling features with a reasonable accuracy at a significantly lower computational cost compared to popular mixed pooling architectures.

On the other hand, with the second mixed pooling strategy, we proposed a novel pooling layer that resides in mixing max and average pooling operations besides using dropout as a probabilistic function in one single layer, we named it mixed-pooling-dropout. Dropout in our proposed method selects activation units based on a Bernoulli distribution regarding a retaining probability $p$ in training time. While in test time, a mean-network strategy was adopted given its efficiency.

Our proposed method was presented under two versions. We compared our proposed method (both models) to conventional pooling methods which were overtaken by our method
with a significant margin. Then, we showed the superiority of our method to max-pooling-dropout since it is the only method in the literature that uses the same strategy as in our method. Moreover, we tried using different retaining probability values to know for sure the optimal value where our method can reach the highest accuracy. We found that $p = 0.5$ is the optimal value. $MiPD$ is a mixture of max and average pooling with $1 \times 1$ convolution for computation optimization purposes with the inclusion of dropout function for stochasticity. Our experiments showed that our proposed method outperforms max-pooling-dropout for different retaining probability value.
Conclusion

In this part, we investigated and introduced some of the feature extraction methods for medical image classification using CNNs. In the first chapter, we performed a comparative study of four different feature extraction methods based on deep neural networks for lung nodule detection. Experimental results show that CNN outperforms RBM, PCA and 2D-DFT, with 2D-DFT being very close to CNN in terms of accuracy but suffers from both high bias and high variance which leads to overfitting and affects the inference phase. The high classification accuracy reached after feature extraction with CNN is an evidence that CNN is very successful in extracting high-level features from CT scans of lungs for the purpose of lung-nodule detection.

In the second chapter, we provided two different mixed pooling strategies. The first one is called fully mixed max average pooling, where we performed a fully mixture of both max and average pooling operations in one single pooling layer with a $1 \times 1$ convolutional layer to reduce both the number of parameters and the computational cost. To examine the performance of this architecture, we adopted the lung image database consortium image collection LIDC-IDRI, for lung nodule detection. And, we compared our architecture to similar mixed pooling strategies, where we showed that with our proposed method, we are 2.68 times faster than any of the mixed pooling strategies adopted in our study.

The second proposed architecture is called mixed pooling dropout, in which we included a dropout function in the mixed pooling layer. Dropout in our proposed method selects activation units based on a Bernoulli distribution regarding a retaining probability $p$ in training time. While in test time, a mean-network strategy was adopted given its efficiency. Our proposed method was presented under two versions, and with both versions we managed to outperform max, average and stochastic pooling in a first experimentation and to also outperform max-pooling-dropout in a second experimentation regardless of the retaining probability values.

We followed the same experimentation strategy followed in the first part; we tested existing feature extraction methods that exist in the literature to perform medical image
classification. Then, we proposed our own architectures and we outperformed conventional and state-of-the-art pooling methods in medical image classification.
General Conclusion

Summary

The main goal of this research thesis was to investigate thoroughly some of deep learning architectures that are suitable for medical image processing, and develop new architectures that improve the accuracy when dealing with medical images.

In the introduction, we rose some research questions to which we are answering as follows:

1. **Q**: Data preparation is one of the mandatory steps to achieve satisfactory results; can a conventional deep learning method perform an efficient segmentation by relying on good data preparation?

   **A**: In chapter 2, part II, we performed a lung CT image segmentation using the U-Net architecture. However, before starting the training process, we had to prepare our own data set to fit the network. The LIDC-IDRI data set contains raw lung CT scans without any segmentation maps. Hence, our first challenge was getting input images from the LIDC-IDRI data set containing lung images, an provide their corresponding segmentation masks manually. It was a time-consuming task. Nevertheless, the nature of the adopted U-Net in dealing with small data sets, was a major advantage to still reach satisfactory segmentation results. Besides, data augmentation also provided more training data. Therefore, the data preparation process and data augmentation had a huge impact on ameliorating the performance of lung CT image segmentation using the U-Net architecture.

2. **Q**: Feature extraction is one of the main steps to prepare data for any computer vision system, it consists of extracting only relevant characteristics from data for next phases. What are the feature extraction methods that can help our deep learning network achieve the best segmentation result?

   **A**: In chapter 3, part II, we proposed a new deep learning architecture for brain tumor semantic segmentation based on some of the state-of-the-art deep learning feature
5.5 Conclusion

extraction methods. These feature extraction methods were organized in a multi-scale way, then followed by an attention mechanism to put more focus on the brain tumor. With the inclusion of these feature extraction methods into our proposed architecture, we managed to keep different kinds feature characteristics for the attention mechanism to work with. Therefore, we’ve reached the best semantic segmentation results (relying on robust feature extraction methods) compared to U-Net, Attention U-Net and FCN.

3. Q: CNNs are one of the state-of-the-art algorithms that made a huge impact in deep learning raise, it consists of a variety of feature extraction functions followed by a classifier. Medical image classification also relies on good feature extraction methods to reach higher accuracy, thus CNNs can be a good fit for that matter. Are convolution and pooling methods the best choice to optimal features extraction for CNNs? What are the enhancements we could make to CNN’s features extraction methods to improve its performance?

A-1: In chapter 4, part III, we presented a deep learning feature extraction comparative study. In this study, we differentiate between local and general feature extraction methods and we focus more on the general feature extraction methods, since CNNs are considered general. Thereafter, we gathered some popular general feature extraction methods to compare their performance to the feature extraction part of a standard CNN. In the end, we came to conclude that convolutions and poolings are the best match for a CNN to achieve a satisfactory classification accuracy.

A-2: In the first section of chapter 5, part III, we proposed a new pooling strategy to enhance the performance of CNNs for image classification. Our proposed pooling method resides on mixing max and average pooling in an optimal manner. The comparison of our method with conventional pooling methods showed that we have made an improvement accuracy-wise. On the other hand, the comparison with other pooling methods that involve mixture of max and average pooling, showed that we have made an enhancement in terms of time performance.

A-3: In the second section of chapter 5, part III, we proposed a new enhancement of the pooling layer in a CNN architecture, in which we extended the mixed-pooling strategy proposed before, by involving a dropout function in order to lower the computational cost and of course we improved the accuracy to outperform conventional pooling strategies beside those involving a mixture strategy. In this work, we gathered our own findings that managed in one way or another to improve the performance of certain deep learning architectures in medical image analysis. There are other methods that were not mentioned in this work, which didn’t show what we were expecting. However,
they still helped in finding other ways to proceed in our direction and improve our work.

We started this thesis in 2016, at that stage, deep learning in computer vision was still not as explored as in the present time. Which made us face some struggles, such as lack of prepared data sets, especially in the medical domain and lack of computational power. Nonetheless, we managed to reach the goal that we set, by preparing our own data sets and finding new way to train our models even with insufficient computational power, also proposing new deep learning architectures after exploring and comparing a lot of architectures.

**Future work**

Our research work has left some uncovered roads, due to lack of prepared data or limited computational resources. However, we have reached some satisfactory results that may lead to some other findings, from which we care to mention these few ones:

As we mentioned before, medical image processing is mainly manifested via detection, segmentation and classification. We only explored the last two phases due to their importance in delivering a decent diagnosis. Nevertheless, we intend to extend our research to touch the detection phase, by proposing improvements on the YOLO [112] algorithm and adopt it for a real time brain tumors detection.

In the work presented in chapter 5, part III, we proposed a new pooling strategy that involves using a multinomial distribution in order to select activations. However, we thought of using different kinds of distribution to select activation and compare their impact on the selected activations, thus on the accuracy.

As for the application of our proposed semantic segmentation architecture, we intend to generalize it to other types of diseases to see if it will perform the same way as in segmenting brain tumors. If not, we want to explore our options in how to make it more generic to perform in whatever types of image data sets.

**Publications**

• Brahim AIT SKOURT, Nikola S. NIKOLOV, and Aicha MAJDA. "Fully Mixed Max-Average Pooling for Convolutional Neural Network" accepted in international network of research and training inrt agadir 2022.

• Brahim AIT SKOURT, Abdelhamid EL HASSANI, and Aicha MAJDA. "Mixed-pooling-dropout for convolutional neural network regularization." Journal of King Saud University - Computer and Information Sciences 34.8 (2022): 4756-4762.


References


References


References


