جامعة مولاي إسماعيل
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵎⵓⵍⴰⵢ ⵉⵙⵎⴰⵄⵉⵍ
UNIVERSITÉ MOULAY ISMAÏL

كلية العلوم
ⵜⴰⵖⵣⵓⵜ ⵏ ⵜⵣⵔⴼⵜ
FACULTÉ DES SCIENCES

Filière : Sciences de la Matière Physique

MÉMOIRE DE PROJET TUTORÉ

Présenté

Par:

Ayoub EL MHAMDI
youssef MADANE

UTILISATION DU DEEP LEARNING POUR IDENTIFIER LES NODULES PULMONAIRES CANCÉREUX SUR LES IMAGES DE TDM

Soutenu le **/07/2023 devant la Commission d'Examen :

Pr RAJAE Sebihi	**Encadrant**

REMERCIEMENTS

Nous tenons à remercier d'abord toutes les équipes pédagogiques de **la Filière Science de la Matière Physique** de la Faculté des Sciences à Meknès, ainsi que les professeurs ayant contribué activement à notre formation.

Nous profitons de cette occasion pour remercier vivement notre Professeur **RAJAE SEBIHI** qui n'a pas cessé de nous encourager tout au long de l'exécution de notre Projet de Fin d'Études, ainsi que pour sa générosité et ses compétences en matière de formation et d'encadrement. Nous lui sommes reconnaissants pour ses aides et conseils précieux qui nous ont permis de mener à bien le présent projet.

Nos vifs remerciements vont aussi aux membres de jury pour avoir accepté de juger ce travail.

A la même occasion, nous voudrons également remercier chaleureusement nos parents qui nous ont toujours encouragés durant notre cursus de formation.

Enfin, nos vifs remerciements sont adressés à toutes ces personnes qui nous ont apporté leur aide précieuse et leur soutien inconditionnel. ■

TABLE DES MATIÈRES

RÉSUMÉ. 4
INTRODUCTION GÉNÉRALE. 5
RÉFÉRENCES BIBLIOGRAPHIQUES. 6

Chapitre I. APERÇU SUR LES SYSTÈMES FRACTALS 7

Chapitre II. Detecting lung cancer using PyTorch 8
II.1. introduction 8
II.2. Approach 9
II.3. Manipulation the Data 9
1. Data Conversions 9
2. Data loading 9
3. Raw CT Data Files 10
4. Training and validation sets 10
5. Loading individual CT scans 10
6. Data Ranges and Model Inputs 11
7. The Patient Coordinate System 11
8. CT Scan Shape and Voxel Sizes 11
9. Converting Between Millimeters and Voxel Addresses 11
II.4. Training 11
1. Overview 12
2. Segmentation 13
3. Predicting 14
4. Save Model 14
5. Classification 15
II.5. Conclusion 15

RÉSUMÉ.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum.

INTRODUCTION GÉNÉRALE.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum.

RÉFÉRENCES BIBLIOGRAPHIQUES.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut.

Chapitre I.

APERÇU SUR LES SYSTÈMES FRACTALS

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum.

Chapitre II.

Detecting lung cancer using PyTorch

II.1. introduction

The project is to create a detector for lung cancer using ct scans. Automating this process will provide an experience in dealing with difficult scenarios where solving problems is challenging (exemple ...).

On choose to use the detection of malignant tumors in the lungs using only a CT scan of a patient's chest as input to illustrate how to overcome technical issues. Automatic detection of lung cancer is challenging, and even professional specialists face difficulty in identifying malignant tumors. Automating the process with deep learning will be more demanding and require a structured approach to succeeding.

Detecting lung cancer early is essential for increasing the patient's survival rate, but it's tough to do manually, especially on a large scale.

The problem space of lung tumor detection is important because it is an active research area with promising results. However, it is also unsolved, which satisfies the authors' objective of using PyTorch to tackle state-of-the-art projects.

In large-scale project, it will be working with 3D data and require data manipulation, as no pre-built library is available for suitable training samples. The project will involve using convolutional layers followed by a resolution-reducing downsampling layer. To handle the computational requirements, you will need access to a GPU with at least `8 GB` of RAM or `220 GB` of free disk space for raw training data, cached data, and trained models.

Instead of analyzing the entire CT scan, it will break down the problem into simpler tasks. CT scans are 3D X-rays consisting of a three-dimensional array of single-channel data, with each voxel having a numeric value that approximately corresponds to the average mass density of the matter contained inside.

As for choosing the batch size, it depends on your specific situation. For example, with an image size of 2400x2400x3x4, a single image takes 70 MB, so

a batch size of 5 might be more realistic. However, this depends on the available GPU memory, and using 16-bit values instead of 32-bit can help double the batch size [Source 3](https://ai.stackexchange.com/questions/3938/how-do-i-handle-large-images-when-training-a-cnn).

II.2. Approach

The goal of this project is to create an end-to-end solution for detecting cancerous nodules in lung CT scans using PyTorch. The approach involves five main steps:

1. Loading the CT data and converting it into a PyTorch dataset.
2. Segmenting the image to identify potential tumors.
3. Grouping interesting voxels to form candidates.
4. Classifying the nodules using a classification model.
5. Diagnosing the patient based on the malignancy of the identified nodules, combining segmentation and classification models for a final diagnosis.

II.3. Manipulation the Data

II.3.1. Data Conversions

To process the data, it is necessary to convert raw data files into a format that is usable by PyTorch, which means converting the row data from 3D array of intensity data to `Tensors` pyTorch format. This data is around 32 million voxels, which is much larger than the nodules. To make the task more manageable, the model will focus on a relevant crop of the CT scan. There are various steps involved in processing the data, including understanding the data, mapping location information to array indexes, and converting the CT scan intensity into mass density. Identifying the key concepts of the project, such as nodules, is crucial.

II.3.2. Data loading

In this chapter, we will discuss the first step in creating a neural network for detecting lung cancer using PyTorch: handling the dataset. The goal is to produce a training sample from raw CT scan data and a list of annotations. The process is described as transmuting the raw data into the stuff that the neural network will spin into gold.

This heading covers the following topics:

- Loading and processing raw data files
- Implementing a Python class to represent the data
- Converting the data into a format usable by PyTorch
- Visualizing the training and validation data

Overall, the quality of the data used to train the model has a significant impact on the project's success.

II.3.3. Raw CT Data Files

CT data comes in two files: a **.mhd** file of metadata header information and a **.raw** file of raw bytes. The `CT` class loads these files, processes the information to produce a 3D array, and transforms the patient coordinate system to the index, row, and column coordinates of each voxel in the array. Annotation data from LUNA with nodule coordinates and malignancy flags are also loaded.

The `candidates.csv` file contains information about all lumps that look like nodules, whether they are malignant, benign, or something else. We'll use this to build a list of candidates that can be split into training and validation datasets. The annotations.csv file contains information about some of the candidates that have been flagged as nodules, including the diameter. This information is useful for ensuring a representative range of nodule sizes in the training and validation data.

II.3.4. Training and validation sets

For supervised learning tasks like classification, we need to split our data into training and validation sets. We want to ensure that both sets represent the real-world input data that we expect to see and handle normally. If either set is significantly different from our actual use cases, it's highly likely that our model will behave differently than we expect. This split helps us evaluate and improve the model's performance before we deploy it on production data.

II.3.5. Loading individual CT scans

We need to understand how to load and understand CT scan data, which is usually stored in a DICOM file format. The MetaIO format is suggested for easier use, and the Python SimpleITK library can be used to convert it to a NumPy array. The Hounsfield Unit (HU) scale is used to measure CT scan voxel density, with air at -1000 HU, water at 0 HU, and bone at least 1000 HU.

II.3.6. Data Ranges and Model Inputs

Starting with adding channels of information to our samples. To prevent the overshadowing of the new channels by raw HU values, we must be aware that our data ranges from -1,000 to +1,000. We won't add more channels of data for the classification step, so our data handling will remain the same.

Fixed-size inputs are necessary due to a fixed number of input neurons. We want to train our model using a crop of the CT scan that accurately centers the candidate, making identification easier for the model by decreasing the variation in expected inputs.

II.3.7. The Patient Coordinate System

The candidate center data expressed in millimeters, not voxels. We need to convert our coordinates from the millimeter-based coordinate system (X, Y, Z) to the voxel-address-based coordinate system (I, R, C). The patient coordinate system defines the positive X to be patient left, positive Y to be patient behind, and the positive Z to be toward patient head. The patient coordinate system is often used to specify the locations of interesting anatomy in a way that is independent of any particular scan.

II.3.8. CT Scan Shape and Voxel Sizes

The size of the voxels varies between CT scans and typically are not cubes. The row and column dimensions usually have voxel sizes that are equal, and the index dimension has a larger value, but other ratios can exist.

II.3.9. Converting Between Millimeters and Voxel Addresses

Converting between patient coordinates in millimeters and (I,R,C) array coordinates, we define some utility code to assist with the conversion. Flipping the axes is encoded in a 3×3 matrix. The metadata we need to convert from patient coordinates to array coordinates is contained in the MetaIO file alongside the CT data itself.

In CT scan images of patients with lung nodules, most of the data is not relevant to the nodule (up to 99.9999%). To extract the nods, an area around each candidate will be extracted, so the model can focus on one candidate at a time.

II.4. Training

II.4.1. Overview

- **CNN**: The design of a convolutional neural network for detecting tumors are based on alternative image recognition that can be used as a starting point. This Convolutional neural networks typically have a tail, backbone, and head. The tail processes the input, while the backbone contains most of the layers arranged in series of blocks. The head converts the output from the backbone to the desired output form.
- **Epoch Training**: The epoch is divided into 20193 steps called batches, each containing 256 data points. Once the data is loaded and the training process begins, monitoring the performance of the computing resources is crucial to ensure that resources are being used effectively.
- **Metrics**: displays training and validation metrics in a graphical format, making it easier to interpret the data. We can adjust the smoothing option to remove noise from trend lines if our data is noisy.
  - Recall is the ability to identify all relevant things.
  - while precision is the ability to identify only relevant things.

The logging output are include the precision by including the count of correctly identified and the total number of samples for both negative and positive samples.

- **Overfitting**: Overfitting occurs when a model learns specific properties of the training set, losing the ability to generalize, and making it less accurate in predicting samples that haven't been trained on. For instance. To avoid overfitting, we must examine the right metrics. Looking at our overall loss, everything might seem fine, but that's because our validation set is unbalanced, and the negative samples dominate, making it hard for the model to memorize individual details. To prevent overfitting, we use data augmentation, which involves modifying a dataset by applying synthetic alterations to individual samples, resulting in a new dataset with a larger number of effective samples. Five specific data augmentation techniques are discussed, including mirroring the image, shifting it by a few voxels, scaling it up or down, rotating it around the head-foot axis, and adding noise to the image.
- **Data Automating**: This technique are designed to create new training samples from the existing ones by applying simple transformations. The transformations include shifting/mirroring, scaling, rotation, and adding noise.

II.4.2. Segmentation

The process of segmentation to identify possible nodules, which is step 2 of the project's plan. The segmentation model is created using a U-Net. The objective is to flag voxels that might be part of a nodule and use the classification step to reduce the number of incorrectly marked voxels.

II.4.2.1. Semantic segmentation: Per-pixel classification

Semantic segmentation identifies different objects and where they are in a given image. If there are multiple cats in an image, semantic segmentation can identify each cat's position. The existing classification models can't pinpoint where the cat is; they can only predict whether or not a cat is present in the image.

Semantic segmentation requires combining raw pixels to develop specific detectors for items like color and then building on this to create more informative feature detectors to finally identify specific things like a cat or a dog. Nonetheless, the segmentation model will not give us a single classification-like list of binary flags like classification models since the output should be a heatmap or mask.

The U-Net architecture is a design for a neural network that can produce pixel-wise output and was invented for segmentation. The design of this architecture is complicated, as it is a lot different compared to the mostly sequential structure of the classifiers.

The U-Net architecture is good for image segmentation. The model has a U-shape, and it operates at different resolutions. It first goes from top left to bottom center through a series of convolutions and downscaling, then uses upscaling convolutions to get back to the full resolution. The key innovation behind U-Net is having the final detail layers operating with the best of both worlds.

II.4.2.2. Updating the dataset for segmentation

Our source data remains the same: CT scans and annotation data. But, our model produces output of a different form than we had previously.

We are going to begin by converting the nodule locations we have into bounding boxes that cover the entire nodule.

To accomplish this goal, we start the origin of our search at the annotated center of our nodule. We then examine the density of the voxels adjacent to our origin on the column and row axis until we hit low-density voxels, which

indicate that we've reached normal lung tissue. Finally, we search in the third dimension.

II.4.2.3. gole of segmentation

The process of designing and creating datasets for training and validating a segmentation model for detecting nodules in CT scans.

- For input into UNet, we have seven input channels; $3 + 3$ context slices, and 1 slice that is the focus for what we're actually segmenting.
- We have one output class indicating whether this voxel is part of a nodule.

II.4.3. Predicting

Create an empty image with 512x512 pixels and three color channels.

The image is used for flagging false positives and false negatives in a prediction. False positives are marked in red and overlaid on the image, while false negatives are marked in green. Pixels that are both false negatives and false positives are marked in orange. The final image is clipped between 0 and 1.

The goal is to have a grayscale CT image with predicted-nodule pixels in various colors. Red is used for incorrect pixels (false positives and false negatives), green is used for correctly predicted pixels inside the nodule, and half-strength mask added green is used for false negatives, which appear as orange.

II.4.4. Save Model

To save only the parameters of the model, this approach allows us to load those parameters into any model that expects parameters of the same shape, even if the class doesn't match the saved model.

II.4.4.1. Writing Code for Nodule Analysis

- Segment CT scan image: Isolate and label regions of interest within the CT scan image using segmentation techniques.
- Grouping: Group the segmented voxels into "lumps" based on their location and proximity to one another. A simple way to do this is to identify the boundaries of each lump, such as by finding the dashed outline surrounding highlighted areas in the image.
- Calculate center of mass (COM) coordinates: Calculate the COM coordinates for each lump of segmented voxels. This provides useful information about the location and distribution of specific features within the image.

- Classification: Use machine learning algorithms to classify the resulting sample tuples containing the COM coordinates, based on criteria such as size, shape, and alignment. This can help identify specific structures or patterns within the image and provide insights into the underlying condition being examined.

![figure 14.3]
(https://cdn.mathpix.com/cropped/2023_04_27_14ff7830b8aaf17904d3g-177.jpg?height=972&width=1430&top_left_y=189&top_left_x=223)

II.4.5. Classification

This step involves dividing the CT scan into individual slices. The output of the segmentation step is an array of per-pixel probabilities, indicating whether the pixel is part of a nodule. These slice-wise predictions are collected in a mask array with the same shape as the CT scan input, and a threshold is applied to the predictions to obtain a binary array. A cleanup step which shortens the flagged area and removes small components.

II.5. Conclusion

The identifying nodule candidates in CT scans for possible cancer detection. A connected-components algorithm is used for grouping the suspected nodule voxels. The labeled chunks are passed on to a classification module to reduce false positives. Finally, the identified regions in the CT scan are cropped and passed onto the classification module using DataLoader.

We use a data loader to loop over a candidate list to threshold the output probabilities to get a list of things our model thinks are actual nodules, which would be output for a radiologist to inspect while adjusting the threshold to err a bit on the safe side. A single CT scan from the validation set is run and 16 nodule candidates are found.

The task of identifying malignant nodules from benign ones in CT scans after implementing the nodule-detection task of the LUNA challenge. Even with a good system, diagnosing malignancy would need a more comprehensive view of the patient, additional non-CT context, and a biopsy instead of just looking at a particular nodule in isolation on a CT scan. This task is likely to be performed by a doctor for some time to come.

The key takeaways from the article are:

- Splitting training and validation (and test) sets between patients is important to avoid errors.
- Converting pixel-wise marks to nodules can be achieved using traditional image processing.
- The diagnosis performs both segmentation and classification.
- TensorBoard can help us visualise and identify network anomalies.
- There is no magic bullet when training neural networks.