Clément BARAILLE

Biomed
2023-2024

ORANGE LABS, 28 Chemin du Vieux Chêne,
38240, MEYLAN, FRANCE

Artificial learning, classification and AI using Maxwell platform

From 05/02/2024 to 30/07/2024

Under the supervision of:
- Internship supervisor: Joël Gardes, joel.gardes@orange.com
- School supervisor: Alice Caplier

École nationale supérieure de physique,
électronique, matériaux
Phelma
Bât. Grenoble-INP – Minatec
3 Parvis Louis Néel – CS 50257
F-38016 Grenoble Cedex 01

Tél +33 (0)4 56 52 91 00
Fax +33 (0)4 56 52 91 03
http://phelma.grenoble-inp.fr

Acknowledgments

First and foremost, I would like to express my sincerest gratitude to my internship supervisor, **Joël Gardes**. He spent a lot of time to bring me the background required to complete my missions and allowed me to discover the sophisticated field of the artificial intelligence through a very interesting approach. He made himself very available and always helped me as soon as I needed. He has always passed on his knowledge with passion, which made this internship all the more rewarding. To top it all off, thanks to his warm and friendly attitude, he was always very pleasant to deal with.

I would also like to warmly thank **Christophe Maldivi**, who was a great support to me during this internship. On the one hand, he was there to help me integrate some good programming practices when I first arrived, but he also gave me invaluable advice throughout the whole experience. In this way, he helped me to be more efficient, but also contributed to making my missions much more enjoyable.

Then, I would like to thank **Jacques Demongeot**, who passionately passed on to me the knowledge I needed to successfully complete my first assignment. Through his words of guidance, he has made a major contribution to my biological culture, enabling me to better grasp some of the concepts involved in this internship.

Finally, I'd like to thank the rest of the people I interacted with at Orange Meylan, who made this internship so rewarding.

Contents

Glossary

Acronyms and abbreviations:

- AI: Artificial Intelligence
- ML: Machine Learning
- DNA: Deoxyribonucleic Acid
- RNA: Ribonucleic Acid
- "5P" medicine: medical approach meeting which is preventive, predictive, participative, personalized and proven
- DL: Deep Learning
- NCBI: National Center for Biotechnology Information
- NLP: Natural Language Processing
- AL: Archetypal Loop
- ANN: Artificial Neural Networks
- MSLD: Monkeypox Skin Lesion Dataset
- PCA: Principal Component Analysis
- CNN: Convolutional Neural Network
- API: Application Programming Interface
- TDD: Test Driven Development

Mathematical notations:

- x_i: Input signal i for a neuron
- w_i: Weight attributed to the input signal x_i
- b_i: Bias associated to the input signal x_i

Technical terms:

- Phylogenetic tree: Graphical representation that shows relations between species.
- Pentamers: Pattern of 5 nucleic bases whose frequency is used in a genome classification approach.
- Overfitting: An error that occurs in Machine Learning when a model is too closely aligned to a limited set of data, leading to disability to generalize its learning.
- Transfer Learning: Machine learning technique in which a pre-trained model is used on another task that for which it was initially designed.

List of figures

List of Tables

1 Overview of the company

Orange Innovation/Marketing & Design/Xperience Design Lab

Orange Innovation/Marketing & Design/Xperience Design Lab (formerly Orange Labs Services until June 1, 2021) is an entity at the core of the Orange Group responsible for innovation in IT services. Historically part of France Telecom's "research & development" division, its aim is to develop the company's strategy, research and enable the company to remain at the technological forefront in the fields of software and IT services with a view to designing telecommunications products and services for both the general public and business players. More broadly, it is part of Orange Innovation (formerly Technology and Global Innovation), which reports directly to Orange General Management, in charge of innovation. It includes other entities notably in research or networks.
This internship was carried out at the Orange site in Meylan, bringing together teams from the Technical and Information System Department (reporting to Orange France) and Orange Innovation teams. This entity is organized into thematic departments (Home services, Commerce & Mobile Banking, . . .) focused on markets and transversal departments centered on skills (Architecture Enablers and Security, Integration Validation Automation).

Reporting team

For this internship, I'm attached to the XDlab Grenoble team, which counts 15 members. The team's missions are centered on the study of solutions for instrumenting user experience and design. Along the way, I was supervised by people both inside and outside Orange:

- Joël Gardes: Researcher, Dr.
- Christophe Maldivi: Software Development Engineer
- Jacques Demongeot: Professor Emeritus at the Grenoble Faculty of Medicine

Project

At Orange, the organization of research activities is steered by Orange Labs Research. These activities are divided into 9 research areas, which in turn are divided into research programs that are subdivided into research projects. My internship took place in the "AI for Health" project (the application of artificial intelligence tools for health) steered by Hervé Provost. This project is itself part of the "Sustainable Digital Services," one of the research programs of the "Solutions for a Sustainable Economy" research area.

2 Introduction

2.1 The role of artificial intelligence in biology

In the last few years, artificial intelligence (AI) has known an exceptional evolution, integrating more and more into our daily lives, and especially in professional environments. Indeed, it is no longer used exclusively in research sectors, or in information technologies, but now extends to many fields such as commerce, finance, or education. This expansion is simply explained: many sectors require the use of large volumes of data that can be processed by machine learning (ML) algorithms. The latter are able to extract models, allowing among other things to perform classification tasks, establish predictions, detect anomalies or generate content.

Biology is not excluded: it also benefits greatly from advances in AI. Indeed, some biologists use machine learning techniques to overcome some limitations which are inherent in traditional methods. In this way, they can now improve the accuracy and speed of their research.

Some of the best-known applications in biology include genomic data analysis. Genome sequencing generates large amounts of data, requiring tools adapted to these volumes that allows, for example, to identify certain essential patterns, or to predict the order of nucleotides in a DNA molecule.

To name just one, a widely used AI-based bioinformatics tool is AlphaFold, developed by Google DeepMind [1]. It allows to predict the 3D structure of proteins from their amino acid sequence. Even if the scope of application of such a tool may seem relatively limited, it actually improves the understanding of certain fundamental molecular mechanisms, promotes enzyme engineering which is widely used in some industrial processes in favor of the environment, or improves the quality of products from the food industry.

Thus, the AI integration in biology perfectly shows its potential to revolutionize various scientific fields, including in medicine where its use has already redefined diagnostics and treatments.

2.2 The role of artificial intelligence in medicine

Today's medical community is moving towards "5P" medicine: preventive, predictive, participative, personalized and proven. This innovative approach to healthcare emphasizes proactive, patient-centered management. Artificial intelligence can contribute to each of the fundamental principles of "5P" medicine. - In preventive medicine, for example, AI is used to identify risk factors and prevent certain diseases before they occur. Indeed, some data analysis applications identify trends and risks for certain infectious diseases. - The predictive aspect is ensured by AI's ability to predict the onset of pathologies based on the data at its disposal. Among the many applications, some are capable of predicting the risk of heart attack 5 years in advance [2] or the risk of breast cancer [3]. - AI also makes it much easier for patients to play an active role in their care. This is made possible by applications that track symptoms and facilitate data sharing and analysis with doctors, such as MySugr, an application that complements the monitoring of type 1 and 2 diabetic patients [4]. - Medicine is also becoming increasingly personalized thanks to AI. On the one hand, it can analyze genetic data to recommend targeted therapies that depend on the patient, and on the other, it can adjust drug doses according to analyses of responses to treatment. This latter aspect can be illustrated, for example, by the Watson for Oncology system, which offers tailored treatments to lung cancer patients [5]. - Finally, AI helps to prove that a treatment is effective by collecting and analyzing clinical trial results and patient responses. In this way, it is possible to identify trends or significant results, revealing the effectiveness of a treatment.

Thus, AI is a significant aid to medicine, improving diagnostic accuracy and optimizing clinical trials. It also enables targeted therapies to be recommended, making certain treatments more effective. Nevertheless, it is essential to moderate these advances: the emergence of AI can deteriorate the relationship between patient and doctor, which is sometimes essential to ensure proper follow-up. What's more, the use of these tools often requires large databases, and calls for particular attention to confidentiality. Finally, particularly in this environment, traceability is essential: in the event of a fault, it is necessary to be able to identify the source. However, with the emergence of Deep Learning (DL) models, which have a "black box" aspect, it is not necessarily easy to identify the source of errors.

This is why we need to continue investigating new tools capable of overcoming these problems. The development of the Maxwell platform, presented in the following paragraph, is part of this approach.

2.3 Introduction of the Maxwell platform

In recent years, my tutor has been working on the Maxwell clustering platform. Originally designed to meet a request from Orange for a tool to classify multimodal digital data, Maxwell proved to work remarkably well on medical data.

We first recall that in machine learning, clustering is an unsupervised clustering technique that aims to divide a dataset into homogeneous subsets called clusters. Each cluster groups objects that share similar characteristics, while being distinct from objects belonging to other clusters.

Maxwell allows the classification of heterogeneous contents, in an agnostic way (i.e. without referring to an ontology of the data but by directly exploring the computer coding of the contents), and unsupervised (without prior indication of the number of classes expected, while including a strategy of autocorrection of classifications made according to purely statistical criteria discovered by the system itself during processing operations).

One of Maxwell's key features is that the clustering is performed directly on byte sequences. As a result, this tool offers a wide range of applications, from text files to multimedia content. An example of this clustering is illustrated in Appendix 1.

Maxwell's clustering method is based on the comparison of several digital objects, using fundamental concepts from information theory [6]. By considering that all digital objects can be represented by a sequence of bits worth 0 or 1, we can evaluate their similarity by comparing their coding. These binary elements are universally encoded in bytes. For this purpose, the approach of algorithmic information theory, in particular Kolmogorov complexity, has been explored. The Kolmogorov complexity measure corresponds to the smallest amount of information required to construct an object by an algorithm (i.e., the length of the smallest program for calculating a given random variable). It is expressed in bytes.

If two objects have close Kolmogorov complexities, this suggests that they share common information. However, its implementation in algorithmic terms is not possible because this complexity is incalculable. Indeed, it is impossible to design a program that returns the value of the Kolmogorov complexity for any input, due to the halting problem as described by Rice's theorem. P. Vitányi, R. Cilibrasi, M. Li and C. H. Benett have shown that one way of approaching this complexity is to use lossless compressors. Thus, from compressed representations of two objects and Burrow-Wheeler transforms, we can deduce a normalized compression distance which, thanks to Bennett's equation, turns out to be an information distance. On the one hand, the use of Burrow-Wheeler, which is the entry point of any lossless compressor, guarantees the reversibility of the transformation: it is possible to start from the transform to reconstruct the sequences. On

the other hand, since compressors are lossless, reversibility applies to all stages. In this way, the expression of the completeness property guarantees that the Kolmogorov approximation is correct, and thus that data fidelity and integrity are maintained throughout the process. The algorithm is globally simple, with low computational complexity, but requires a sorting step (Dual-Pivot Quicksort) which is the most laborious part of the process due to its relative complexity and the computational time it takes to reorganize the data efficiently. To clarify any possible misunderstandings, an example of how to calculate this distance between two words, "badinage" and "baignade" -both anagrams-, is presented in Appendix 2.

Since the manipulated quantity corresponds to a distance, it is possible to deduce the following properties, which are essential for clustering:

- The identity : $d(x, y) = 0 \iff x = y$
- The symmetry : $d(x, y) = d(y, x)$
- The triangular inequality : $d(x, y) \leq d(x, z) + d(z, y)$

$x, y, z \in X$, X being a metric space It is important to add that the compressor initially used was based on the Huffman algorithm, which compresses data using shorter codes for frequent characters and longer codes for less frequent ones, but it was neither distributive nor commutative. It was therefore replaced to ensure distance symmetry.

Maxwell then gathers the distances separating each object we want to classify into a matrix, in which we seek to identify homogeneous, isotropic regions. To do this, we perform a triangulation in the distances, based on a current element P1, its nearest neighbor P2 and the nearest neighbor of the latter, P3. This representation forms triangles whose area and isotropy, respectively, can be evaluated using the following formulas:

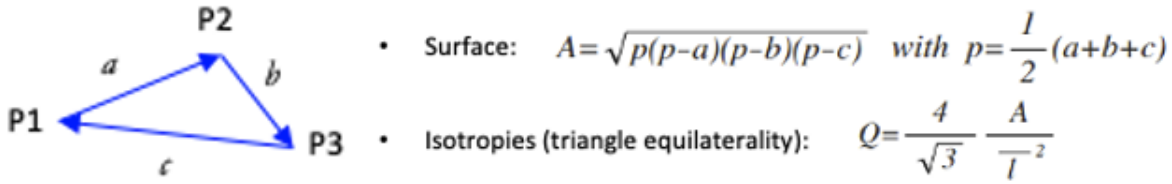


Figure 1: Triangulation representation and useful formulas for qualifying triangles

From these values, the calculated triangles are subjected to decision tests: first, a histogram of the areas is built, then the triangles with the highest densities are extracted, enabling clusters to be deduced.

At this point, to better understand the rest of the explanation, it is important to have a correct representation of the dynamics of Maxwell's processing:

1) We determine a buffer corresponding to the number of contents processed simultaneously. Its size must be such as to provide a representative sample of the problem.

2) Contents are classified in relation to existing clusters. If there are none, we move on the next step.

3) Non-classifiable elements are clustered locally.

4) Other elements are injected into the buffer, and the steps are repeated.

This iterative process is used to construct Maxwell's clusters, which can be defined as follows:

- One or more barycenters: the best representative of a data group (the element(s) with the most connections in the triangle graph)
- A radius: all elements located at a distance from the barycenter less than this radius will belong to the same cluster as the barycenter.

Like some other clustering tools, Maxwell allows you to perform hierarchical clustering (once flat clustering has been completed). This method allows the construction of tree structures, commonly known as dendrograms, which visualize the relationships between different observations in a dataset. When Maxwell determines the barycenters, it sometimes comes across undecidable situations. In this case, the cluster is defined as a "multiset", and creates a "virtual" barycenter which is the concatenation of the actual barycenters. It then calculates the average of the distances of the cluster elements to this virtual barycenter, enabling a simplified representation for classification. Clustering in this way makes it possible to establish a hierarchy on several levels, iterating over the barycenters of each cluster until only one per group remains.

Besides, an essential feature of Maxwell that sets it apart from more conventional classification approaches is its ability to doubt. When the platform is confronted with data that it cannot categorize, it can create a so-called "multi-hypothesis" cluster, grouping together content that is similar but in fact belongs to several classes. In practice, it is possible to provide Maxwell with metadata to verify the semantic consistency of clusters. This approach leads to multi-hypothesis clusters, in which the labeling of elements is distributed across several classes.

Maxwell can also be used to detect and remove duplicates in a dataset, which can be very useful in machine learning tasks [7]. Indeed, the presence of duplicates in a training dataset can introduce a bias on the models designed, and distort the resulting classifications.

Finally, Maxwell presents its results in the form of a CSV file, in which clusters, distances to barycenters and any metadata are listed.

As already mentioned, Maxwell works particularly well on biomedical data, and we have chosen to cite three of the outstanding experiments that illustrate the multimodal aspect of the Maxwell platform:

- Maxwell clustered around 2,000 conventional chest X-rays, separating those from healthy subjects from those with pneumopathy. The platform not only achieved this, but also succeeded in distinguishing between viral and bacterial pneumopathies within the pneumopathy class.
- In a study on using Machine Learning to help detect and predict the progression of sepsis in patients, selected vital signs of 1492 patients were measured over several days [8]. From over 140,000 clinical records stored as CSV files, Maxwell was able to group patients with sepsis according to the status and progress of their pathology using the reconciled values in these records. Finally, Maxwell turned out to be particularly useful in this study by detecting false labels, indexing errors, which could have distorted the results.
- The Maxwell platform has also been tested on a corpus of 250 cardiac recordings including normal heart sounds, cardiac gallops, extrasystoles and murmurs. While audio extracts are normally classified using Fourier spectra, Maxwell is able to classify WAV files directly. The result is a classification very similar to that of the corresponding spectra.

Moreover, among many other applications, Maxwell has also demonstrated itself in the field of genomics. This will be the focus of the first part of the work carried out during this internship.

3 Main achievements during my internship

3.1 Applications of Maxwell in genomics

In addition to the examples given above, Maxwell can also process text files, and while there are many applications for text classification, we will be focusing on genomic data.

3.1.1 Benefits of classification for genomics

In biology, particular attention is paid to the genome, a sequence that describes all the genetic material of a species, encoded in its DNA and RNA. DNA contains the genetic instructions necessary for the development, functioning, growth and reproduction of these organisms, whereas RNA transports this information to the ribosomes, where it is translated into proteins. According to these definitions, genome classification would enable organisms to be identified and differentiated on the basis of their genetic sequences, thus facilitating the understanding of evolution, the detection of genetic diseases, and the development of targeted medical treatments. In concrete terms, this experiment will get rid of the semantic aspect of the genome: DNA will be nothing more than a long word made up of syllables of 3 characters (codons), extracted from an alphabet of 4 letters (”A”, ”C”, ”G”, ”T”, which represent the nucleic bases). By comparing genomes, we aim to reconstruct phylogenetic trees. These are graphical representations of the evolutionary relationships between different species: the more species share similar characteristics at genotype level, the closer they will be in the tree. Similarly, a phylogenetic tree shows when two species diverge. Numerous methods, which rely heavily on phenotype (the set of observable characteristics of an organism), already exist to produce this type of tree, but here we seek to confirm that classification of textual genomes can achieve the same results.

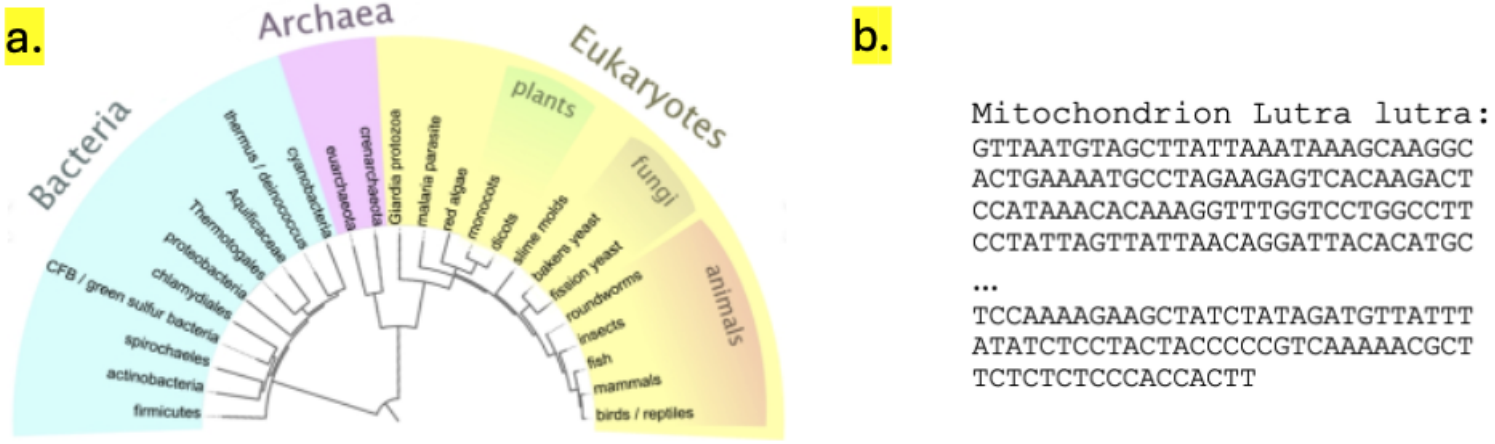


Figure 2: Visualization of the genomic tools used

(a.) Example of representation of phylogenetic tree. (b.) Extract of the mitochondrial DNA of the species Lutra lutra (Eurasian river otter), arbitrarily chosen as an example [9].

In the next section, we will highlight some of the challenges raised by the genome format in classification tasks.

3.1.2 Limits of classical deep learning approaches for genomes classification

Several techniques based on artificial intelligence, more traditional than Maxwell, already exist to classify texts. They form part of the field of application of Natural Language Processing (NLP), a field of research that enables the creation of tools for processing texts and speech. It covers a wide range of applications, including spam detection in e-mails, automatic text translation and even virtual assistants. Text classification using AI follows a rather generic model, based on linguistics, and therefore semantics:

- Firstly, the texts to be classified undergo an initial cleaning stage, during which punctuation, determiners and sometimes even linking words are removed. In this way, only the words most likely to bring meaning to the sentence are retained.
- This is followed by a normalization stage: firstly, the text is separated into several tokens (a token generally corresponds to a word), which then undergo transformations. We can choose to retain only the root of the words, in which case we speak of stemming. Otherwise, we retain only their canonical form (the infinitive for verbs, the masculine singular form for adjectives, etc.); this is called lemmatization.
- The modified texts are then transformed into digital data using various encoding techniques.
- Finally, data can be classified, using previously established rules such as keyword detection, or using classic Machine Learning or even Deep Learning techniques..

This entire process is illustrated in figure 3.

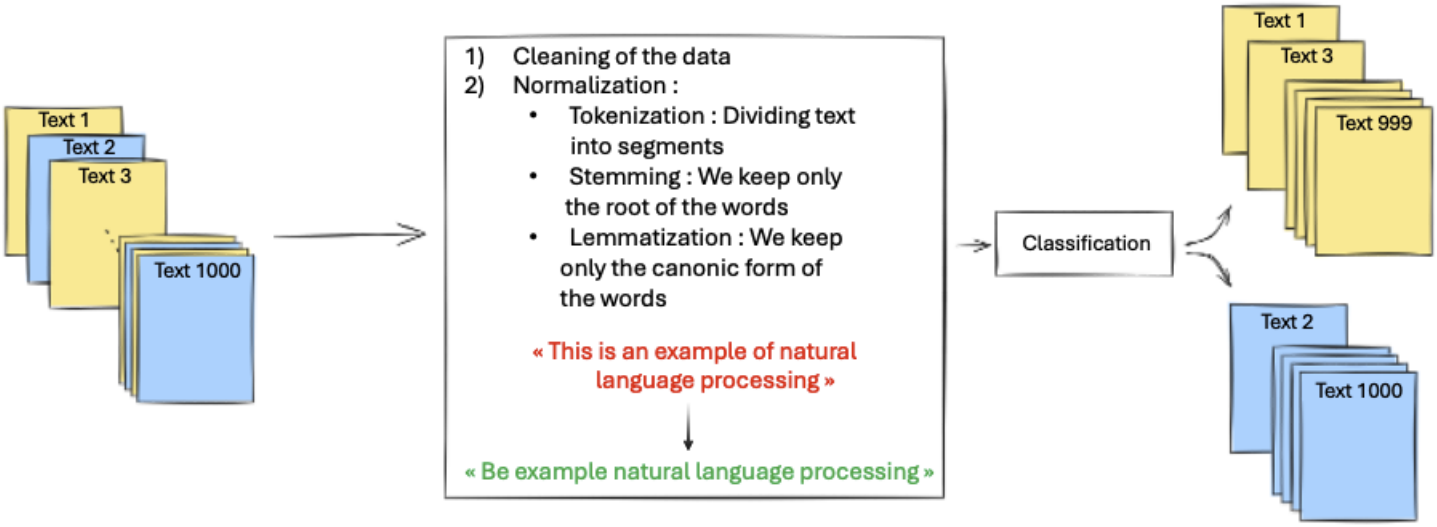


Figure 3: Global pipeline used for text classification

This approach works very well on certain tasks, such as classifying movie reviews [10], but it would not be effective on the long texts represented by genomes. On the one hand, as already mentioned, DNA is a very long word. It is not possible to break it down into tokens. It has been envisaged to split it into 3-character tokens, as codons are, but this would lead to a loss of information, mentioned later. Finally, DNA is subject to numerous mutations: a base can be added to or subtracted from the sequence, or even one base can be replaced by another. These modifications are liable to distort classification.

Fortunately, thanks to the nature of distance calculation, Maxwell was able to alleviate these concerns.

3.1.3 Difficulties in repatriating genomics data

The genomic data we will be manipulating are repatriated from the NCBI site (National Center for Biotechnology Information), an online resource that provides access to a large collection of databases and tools for research in biology and medicine. Nevertheless, their provenance can be a source of various concern. First of all, they are imprecise, since they come from numerous contributors, and depend on the state of progress of sequencing, which means that several genomes can coexist for the same species, for example. This can make analysis more complex, requiring additional checks to ensure the accuracy of the information. On the other hand, the data is often incomplete, with some sequencing not yet completed. This incompleteness can lead to difficulties when using the data, requiring further research to fill in the gaps. Moreover, the data corpus is unstable, and is bound to evolve regularly. Finally, the data can be voluminous: even if only text files are manipulated, the quantity of data to be downloaded can exceed several gigabytes. Managing these data volumes requires particular attention, especially as the NCBI site, in addition to being quite old, imposes a rather strict policy to avoid saturation of their servers. This policy may include limitations on the number of requests or the size of downloadable files, making data retrieval more laborious and time-consuming. To cut a long story short, the complexity of the NCBI database structure can represent a further difficulty, as it is sometimes difficult to navigate and find genomes specific to a certain application.

With these considerations in mind, we designed a graphical interface to facilitate data retrieval.

3.1.4 Conception of a graphical interface

3.1.4.1 Downloading data from NCBI website

First of all, it should be noted that several tools already exist for downloading data corpora more or less easily from NCBI; the best known being the Entrez Programming Utilities program [11], which can be used to search for and retrieve data from NCBI using HTTP requests. During this mission, the aim will not be to reinvent the way of downloading data, but rather to modify and integrate some existing tools in order to design a graphical interface that is very easy to use. Considering that the data will then be used for classification, we would ideally like to download a large quantity of genomes at once. To achieve this, we download genomes by "taxon". In phylogeny, a taxon refers to a group of organisms that share common characteristics. Taxonomy works in a hierarchical fashion, as shown in figure 4: the further down the pyramid you go, the more specific the groups, and the fewer genomes they contain.

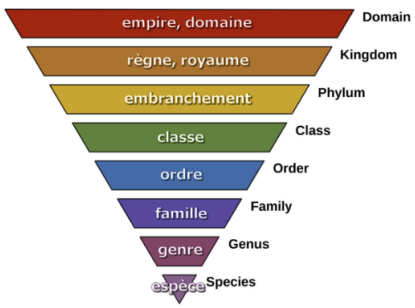


Figure 4: Pyramidal representation of different phylogenetic groupings

(image extracted and modified from the website `https://www.aquaportail.com/`)

Contributors of the NCBI website have developed command lines that allow you to download taxa from the site. In addition to being able to specify the desired taxon, certain attributes can be added to the instruction, including two in particular that we will be using:

- "–reference": by specifying this in the command line, only genomes annotated "RefSeq" are uploaded to the database. This note stipulates that the sequences have been approved, and is therefore a guarantee of reliability.
- "–assembly-level complete": this tag allows to download only completely assembled genomes, i.e. those that offer a complete representation of the genome (regardless of the validation level).

The command lines therefore take the following form: «datasets download genome taxon taxon_name –reference –assembly-level complete»

This command then downloads the genomic data for each species present in the specified taxon. These data take the form of fasta files, which are similar to text files, containing all the different nucleic acid sequences corresponding to the same species. In addition, the sequence name (containing the species name) is written at the beginning of the sequence. To ensure that Maxwell classification is based solely on the sequence, and not on the names entered in the fasta files, a data formatting step is required. To meet this need, we took care to separate the different sequences present in a single fasta file into several text files, for which the nomenclature becomes the file name. In this way, for each species, we moved from a fasta file containing several named sequences to text files associated with a single sequence.

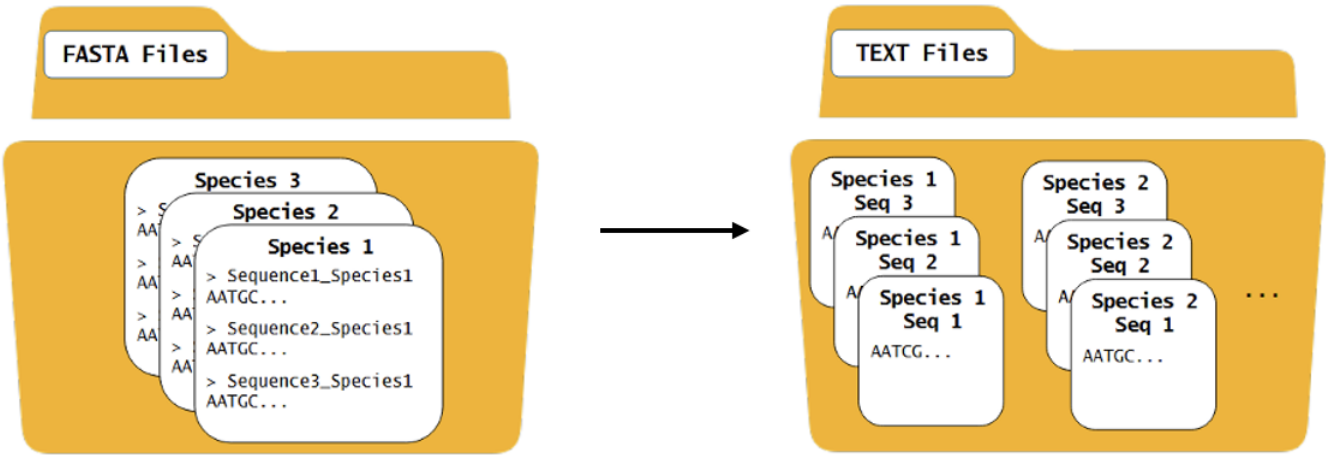


Figure 5: Diagram illustrating the conversion of fasta files into text files

3.1.4.2 The AL proximity, a metric to serve as Maxwell's standard

In parallel with Maxwell, we are working with biomathematician Jacques Demongeot on another way of reconstructing phylogenetic trees that can be used as a benchmark. This method is based on the study of RNA. More specifically, we will be focusing on primordial RNAs, hypothetical forms of RNA considered to be the first genetic molecules capable of storing genetic information and catalyzing the biochemical reactions necessary for life. From these primordial RNAs, we can find particular patterns of 5 nucleic bases, the "primordial pentamers", in the genomes of all

species, which will enable us to trace the phylogeny we are talking about. In fact, the presence of these pentamers in the genome is statistically evaluated using a measure known as the AL (Archetypal Loop) proximity, making it possible to quantify just how close two species are, while at the same time giving us information on their age (the more pentamers, the older the species) [12]. It is precisely because the calculation of this statistic relies on pentamers that a classic deep learning approach using 3-character tokens is not possible. In this way, we can also reconstruct a phylogenetic tree from this metric, which we have compared with Maxwell's results, as shown in figure 6.

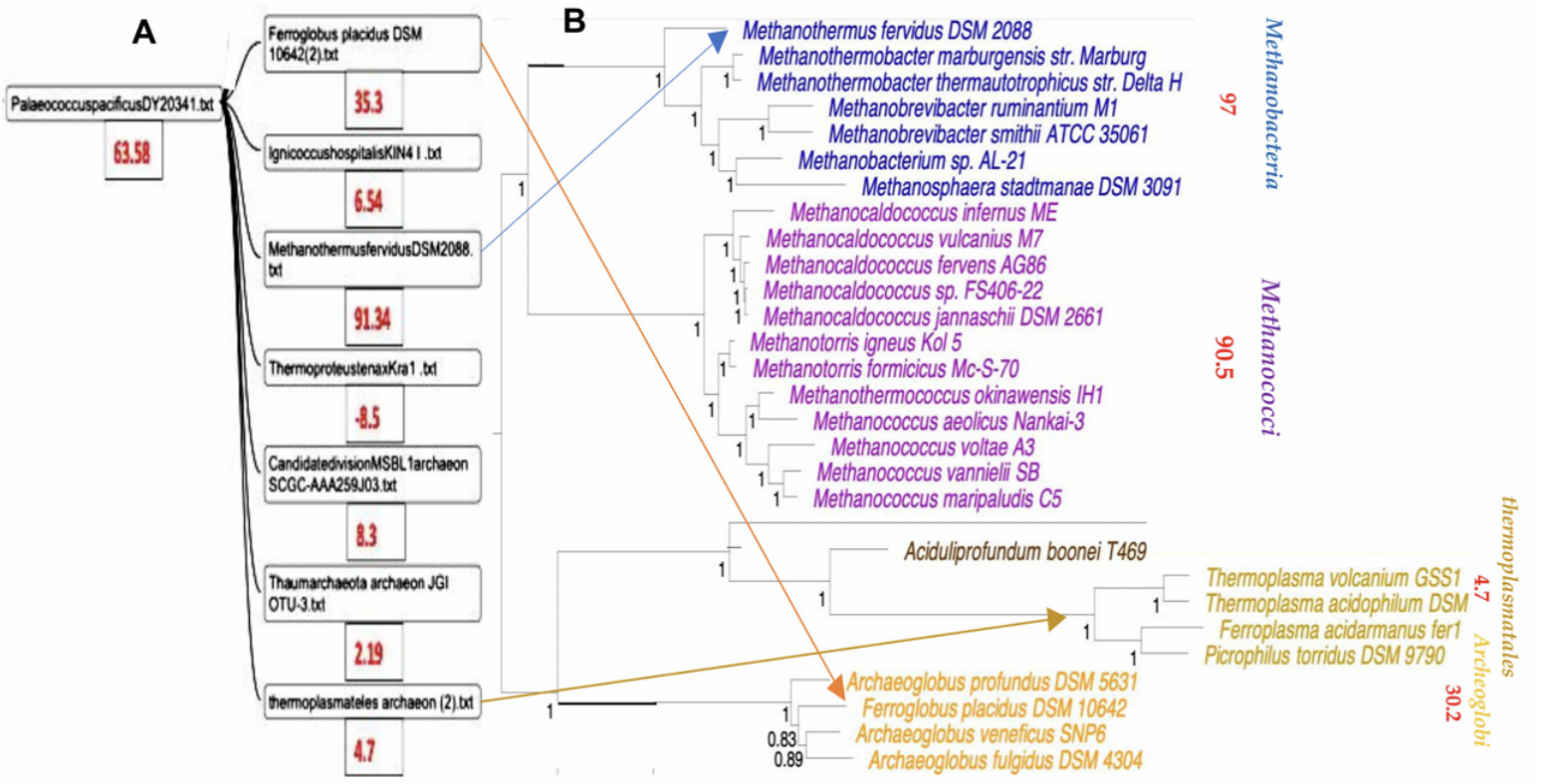


Figure 6: Comparison between Maxwell's results and actual phylogenetic tree

(a.) Phylogenetic tree built by Maxwell. (b.) Part of the archaea phylogenetic tree from [13] with mean AL-proximities (in red)

Here, we can see that Maxwell finds the same clades, the same groupings as with the other approach; but the fundamental difference is that, whereas a tree like this can take several days to build using conventional methods, Maxwell has achieved these results in just a few hours. So, incidentally, in order to be able to keep the AL distance as a standard, calculation tools for data retrieved online have been included in the interface. To do this, we will be adapting an existing code designed by Christophe Maldivi (Orange), who has already worked on this problem before.

3.1.4.3 Graphical interface development

Considering the above, and the NCBI website specifications mentioned above, the graphical interface used to download genomic data for Maxwell must meet the following requirements:

- Allows the user to bypass the command terminal,
- Allow the user to enter the name of the taxon to be downloaded,

- In the event of download failure, indicate the error message,
- Block the download of too many genomes simultaneously,
- Choose whether to download only RefSeq annotated genomes, or only complete genomes,
- Convert between fasta and text files,
- Create html files similar to text files, but highlighting certain statistics required for AL distance calculations,
- Enable fast deletion of downloaded files,
- Generate a csv file for importing data into the Maxwell platform,
- generate a csv file containing the statistics needed to calculate the AL distance for each sequence.

To design a functional interface that meets these criteria, the main Python libraries used are "subprocess" and "tkinter". The former is used to execute command lines from a Python script, while the latter is used to create the interface's graphical appearance. The end result is an ergonomic tool, executable via a shortcut, illustrated in figure 7, which shows different use cases.

Figure 7: Various use cases of the graphical interface

(a.) The download of the taxon "methanococci", with criteria "RefSeq" and "Assembly complete" was successful. (b.) The system does not recognize a taxon name from the NCBI database, download failed. (c.) The taxon euryarchaeota without criteria "RefSeq" and "Assembly complete" contains too many genomes, download failed.

The main difficulty encountered during the development of this interface was setting a limit on the amount of data to be imported. It was necessary to predict how many genomes would be

downloaded, even before initializing the operation. To implement this safeguard, it was necessary to make the messages returned by the terminal interact with the Python code in real time. In this way, it was possible to prevent the collection of data (whose number was recorded in a message returned by a command line) if it exceeded a set threshold value.

As required, this interface enables the genomic data of a taxon to be downloaded and the files used by Maxwell to be generated. In addition, supplementary files facilitating the AL distance study are also provided. Appendix 3 contains two examples of these files, firstly a spreadsheet listing statistics useful for calculating AL distance, and secondly an html file highlighting pentamers and their count.

In the end, the interface appears to be functional so far, but some aspects may still be open to discussion.

3.1.4.4 Discussions and prospects

The interface designed meets the expectations initially mentioned. For instance, it was used to retrieve and classify the genomes of the methanococci species in the article "Information gradient among nucleotide sequences of essential RNAs in an evolutionary perspective". However, while this interface has proved its worth for downloading and formatting fasta data, there is another widely used format, GenBank, which requires different pre-processing. It will therefore be necessary to modify the code behind the interface to support this new format. Furthermore, this interface was designed to run on MacOS, but has been adapted for Windows. However, some elements need to be revised to optimize them according to the operating system used. Finally, NCBI has devised an alternative for downloading large volumes of files, which we decided not to integrate into this interface in the first instance. The first step is to download the zip folder containing the files in a so-called dehydrated format (which is much smaller than the original zip folder), decompress this new version and then rehydrate the data. These steps are performed using command lines, similar to those mentioned above, and could be implemented in the interface quite simply. Nevertheless, for the time being, we have left this aspect aside, preferring an approach in which we download taxa fairly low down in the hierarchy rather than directly downloading the most voluminous, which is more suited to the nature of our work. Finally, this interface is entirely in French, and would require translation if its use were to be extended.

In this section, we have devoted some time to the data retrieval and pre-processing stages, inherent to the artificial intelligence sector. Now, in a second step, we will look at the use of Maxwell in a learning process involving neural networks.

3.2 Hybridization of Maxwell's platform with neural networks

Maxwell and neural networks are both tools that contribute to data classification. Although they use different approaches, we have seen how they can complement each other [14], which I have explored in greater depth during this internship through a series of experiments that I will develop here. First, a few principles of neural networks will be recalled.

3.2.1 Reminders about neural networks

Inspired by animal brain processes [15], neural networks are computer systems that can be used to solve complex problems in fields as diverse as computer vision and natural language processing. The technology is based on information processing, and the resulting autonomous machine learning. In this context, here is how a neuron works [16]:

- On the input side, a neuron receives several input signals $x_1, x_2, ..., x_n$ to which weights $w_1, w_2, ..., w_n$, respectively, are assigned
- It then calculates a weighted sum of these inputs ($z = \sum_{i=1}^{n} w_i.x_i + b_i$) where b_i is a constant used to adjust the neuron's output
- This sum is then evaluated by a non-linear activation function, which can be used to model complex relationships. The image of this function constitutes the neuron's output.

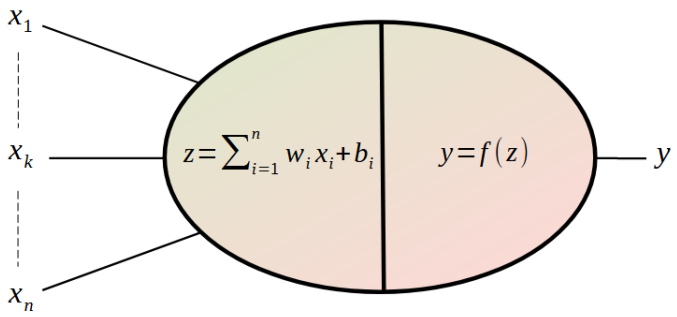


Figure 8: Simplified representation of a neuron, inspired by [16]

Artificial Neural Networks (ANN) models are composed of numerous neurons, organized in layers. As the elements of these layers are interconnected, the complete network can then generate predictions on the data provided as input. A layer in which all neurons are connected to every neuron in the preceding layer is said to be dense. Initially, the first predictions may turn out to be wrong, but it is by repeating the operation of propagating information within the network several times that the model learns to respond to the requested task. This is achieved through the loss function, which quantifies the difference between expected results and the predictions obtained. The gradients of this function are then propagated through the network, enabling the model parameters (w_i, b_i) to be adjusted at the next iteration (epoch) to improve results.

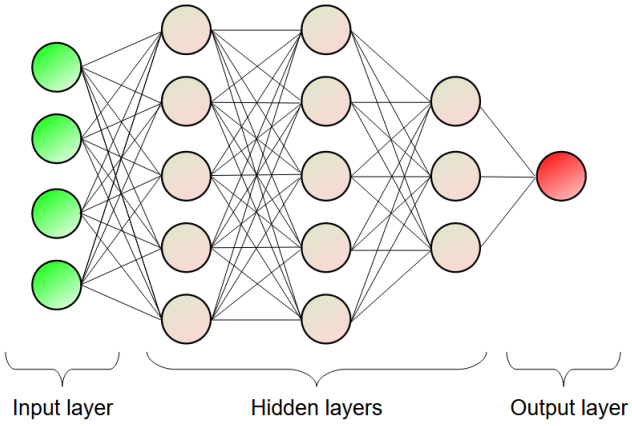


Figure 9: Simplified representation of a neural network

A neural network is therefore structured in 3 blocks: the input layer, which receives the data, a block containing the hidden layers in which the calculations take place, and then the output layer producing the model prediction.

To put this concept into practice, we worked on an image classification task. These are simple files for neural networks to exploit, making this modality a good choice for experimenting with hybridization with Maxwell.

3.2.2 Experimentation on the Monkeypox dataset

As previously mentioned, the aim of this experiment is to measure the impact that the Maxwell platform could have on the training of a neural network. To observe this, we are going to create a model for classifying dermatological images according to two categories: patient suffering from monkeypox (labelled "Monkeypox"), or not (labelled "Other"). This pathology was not chosen randomly: Monkeypox is an emerging disease on the African continent. That is why Jules Tchatchueng of the Yaoundé Pasteur Center approached us as part of the development of an application to reinforce sanitary measures at the borders of Cameroon - a country spared from the disease until now.

3.2.2.1 Introduction of the dataset

The images used for this classification come from the Monkeypox Skin Lesion Dataset (MSLD) [17] [18], designed following the pathology's recent spread to over 65 countries, with the aim of improving early diagnosis. Here is how the global dataset is put together:

- A training dataset: comprising 980 images of patients with the disease, and 1162 images of other dermatological pathologies. It is on this dataset that the neural network will be trained. In this way, at each epoch, it will adjust its weights so as to correctly predict the labels of these images. A special feature of this dataset is that it has been built using a data augmentation procedure. This is a technique widely used in Machine Learning, enabling the size of a dataset to be increased by applying trans-formations to each image. For example, an image can be translated, rotated, zoomed, noised, or even have its chromatic characteristics changed. In this way, we can improve the ability to generalize our network learning to new data.

Another way of visualizing the distribution of this training data is principal component analysis (PCA), which is used to represent similarities and differences between data. Here, it is constructed by projecting the flattened vectors of the images, in a 2D space.

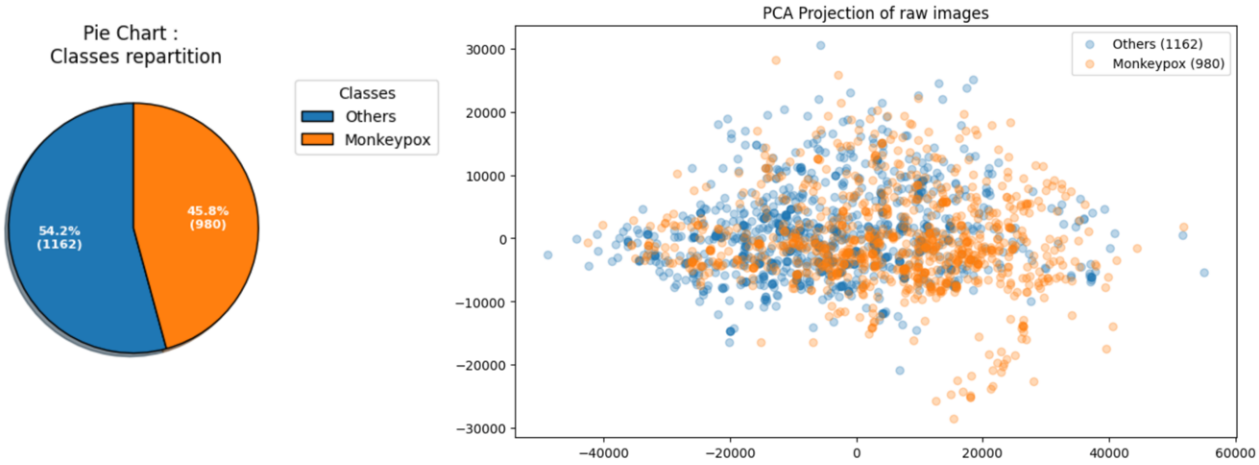


Figure 10: Class distribution of the training dataset

(a.) Pie chart of the class repartition. (b.) PCA diagram

This representation is not essential to the rest of the study, but it does highlight something important: it is very difficult to establish a priori clusters separating Monkeypox images from others.

- A validation dataset: 168 images are of patients coughed up by monkeypox, while 252 others are not. This dataset is used to evaluate the model's performance during training, and to check that the model is not overfitting, i.e. that it is capable of correctly classifying data it has not seen during training.
- A test dataset: 21 photos are those of patients suffering from monkeypox, 24 others correspond to other diseases. Finally, this dataset provides an impartial assessment of the model's performance after training, by estimating its ability to generalize on data it has never seen before.

NB: Currently, another version of this dataset has been developed, MSLDv2.0, which distinguishes between the different pathologies labeled "Others". Since our aim is to improve detection of monkeypox specifically, and not other pathologies, we have not used this dataset and have retained a binary classification.

3.2.2.2 Global methodology of the experiment

To study the impact Maxwell could have on the training of a neural network, we will compare the performance of a model trained on different datasets, deduced from the results of the clustering platform.

- Firstly, the model will be trained and then tested on the sets already established in the MSLD dataset.

- Secondly, the model will be trained only on the best representatives of the dataset, i.e. the barycenters extracted by Maxwell. The model will then be tested on the initial test set, on the complements to the barycenters, and then on the concatenation of these two sets.
- Finally, we will exclude the most ambiguous images from the training dataset, i.e. the multi-hypotheses detected by Maxwell. The tests will then be carried out on the initial test set and on the multi-hypotheses.

In this way, we can attempt to qualify the training dataset. In fact, we can compare the performance of models trained differently, and see whether it is preferable to truncate the initial base. This technique already exists in Machine Learning: it is known as subsampling the training dataset. Several approaches have already proved their worth, such as AdaSelection [19], which optimizes the choice of subsampling to be performed. While this operation is generally carried out in order to save time, we'll be looking here to highlight the dataset that provides the best learning restitution.

3.2.2.3 Results of Maxwell's clustering on the training dataset

The complete training database was submitted to the Maxwell platform. From this clustering, 652 barycenters were identified, and 11 images belonged to multi-hypothesis groupings: these are the images with the most ambiguities. Given the dataset at our disposal, most clusters logically group together an image with its associated augmentations, as shown in figure 11.

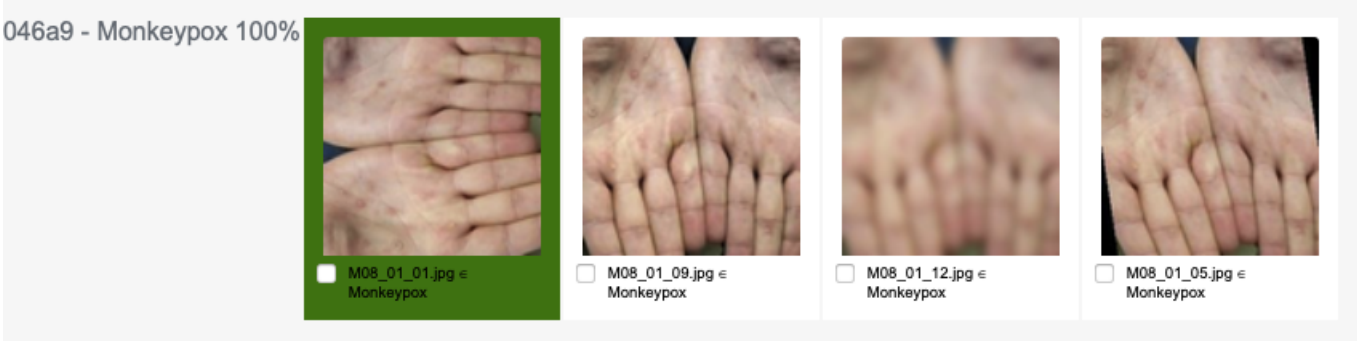


Figure 11: Example of cluster computed by Maxwell

It will then be interesting to see the differences in performance between augmented and non-augmented datasets. Indeed, even if augmentation is normally intended to improve the training of the model, it can also introduce a bias due to image redundancy or introduced noise, and thus complicate the detection of certain dependencies between images.

3.2.2.4 Model building

To classify images using neural networks, we usually use so-called convolutional networks. A convolutional neural network (CNN) takes into account the spatial relationships between pixels in an image through convolution operations. Roughly speaking, instead of using dense layers, a CNN first uses masks that scan all the pixels and their surroundings in the image. In this way, each filter extracts local features such as contours, texture or patterns by analyzing regions the size of the filter. This is followed by a "pooling" stage, during which the dimensions of the features extracted by convolution are reduced. These two steps can be repeated several times, and then at the output of the convolution and pooling layers are a number of dense layers enabling classification according to the extracted features.

To implement these networks, we used Python's Tensorflow Keras library. This is an Application Programming Interface (API) that simplifies the creation of neural networks. We chose this library over PyTorch because it's easy to learn, and frees us from some of the technical details of programming, allowing us to concentrate on design and experimentation.

First of all, we tried out several convolutional network architectures, finally settling on the following :

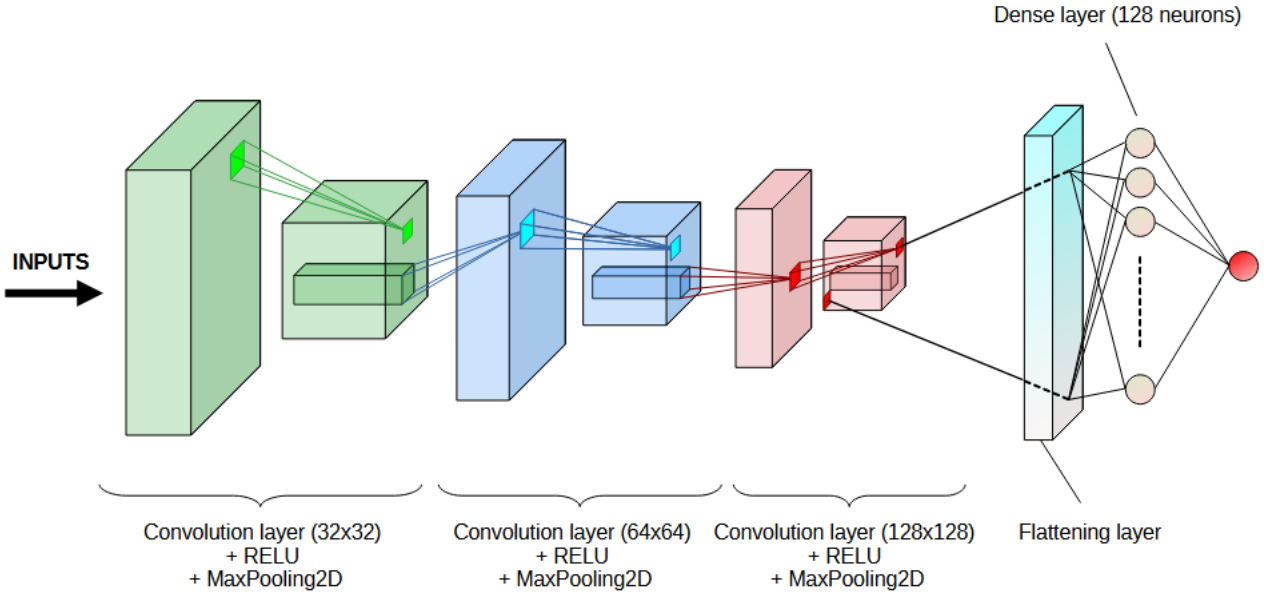


Figure 12: Architecture of the selected CNN

Diagram inspired from a representation on `https://saturncloud.io/`

For this model, at the end of training, set empirically at 20 epochs, with the complete datasets, the classification performances obtained are as follows:

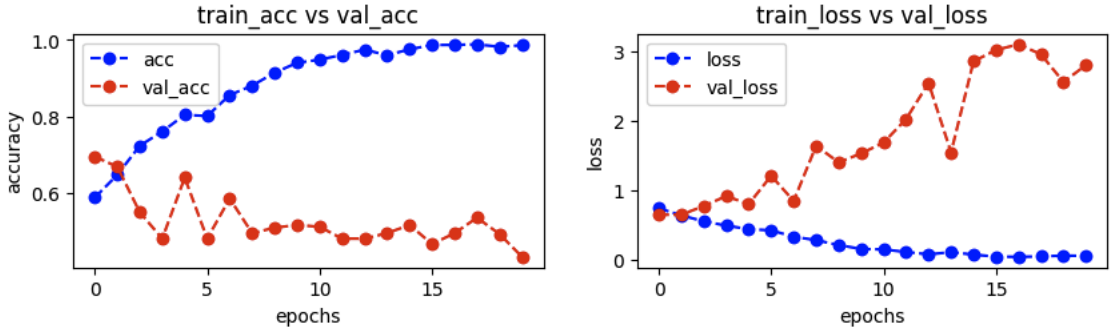


Figure 13: CNN's performances on the initial training dataset

On the left are represented the accuracy curves as a function of the training period; they correspond to the proportion of images correctly predicted by the model during training. On the right, we show the evolution of the loss function during training: we aim to minimize this function for both training and validation. Since accuracy and the validation loss function end up being low and high respectively, these results are unsatisfactory. Here we encounter a classic problem in Machine Learning: we observe a significant difference between validation and training performance,

suggesting overfitting. The model adapts too well to the training data and fails to generalize its predictions to other data.

There are several regularization techniques available to try and overcome this problem, but after consulting several reports of Monkeypox classification projects on Kaggle, we opted for another approach: Transfer Learning.

Transfer Learning consists in using a neural network that has already been trained on a different classification task from the one you wish to perform. Surprising as it may seem, it is possible to use a model designed to classify images of dogs and cats, on dermatological data. All that's required is a slight adaptation of the basic model to its new task. This is based on a fundamental idea of Transfer Learning: if a model is trained on a sufficiently large and varied data set, it can be used as a generic model of the visual world.

In practice, the first layers of the pre-trained network learn to detect fairly general, low-level features, such as edges, textures or colors. The higher up the layers you go, the more specific the features extracted will be. For this reason, there are several possible strategies for customizing a pre-trained model:

- Firstly, you can truncate the basic model. This means removing the last layers of the model that enable classification. In this way, the output of this first model is a features map (representing another image form, easier to classify), rather than a specific class such as "dog" or "cat". We then use a classifier (one or more dense or convolutional layers) to classify these feature maps.
- Alternatively, we can freeze the first layers of a pre-trained network so as to retain its ability to extract general features, and make only its last layers trainable in order to adapt them to a more specific classification problem. It is then possible to add a few more layers to improve the model's adaptability.

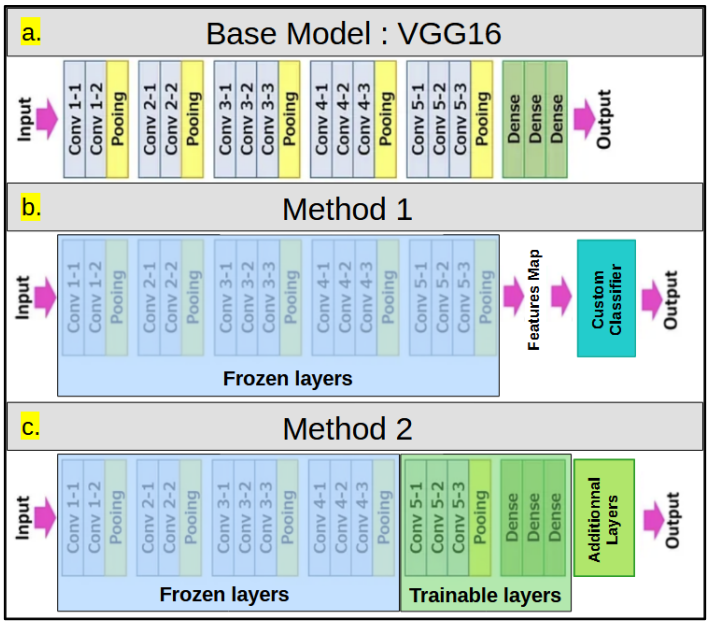


Figure 14: Examples of Transfer Learning patters

(a.) The base model, VGG16, a well-known image classifier widely used in Transfer Learning. (b.) First strategy: features map are extracted from the first layers of the base model, then are classified. (c.) Second strategy: lower layers of the base model are frozen, higher are trainable and supplementary layers are added to refine the classification.

The second strategy was used in these experiments. All that remains to be determined is which basic model to choose, and how many layers to freeze in order to adapt it to Monkeypox image classification. For the sake of practicality, pre-trained models were selected from Tensorflow's Keras Applications library. This library provides popular neural network models, trained on large datasets. Here are the different models tested in this experiment:

- VGG16, known for its high classification performance and simplicity. It's a network capable of extracting a variety of features, making it a very interesting candidate for Transfer Learning [20].
- MobileNetV2, a lightweight neural network designed for mobile and embedded applications. Its structure considerably reduces the number of parameters to be trained, while guaranteeing high performance [21].
- Densenet121, a network characterized by dense connections between layers, enabling better propagation of errors and therefore more efficient adjustment of its parameters [22].
- InceptionV3, named after the Inception modules it uses. These are blocks that combine several convolutions with different filter sizes and several pooling operations. This architecture enables the network to capture visual patterns at different scales [23].
- ResNet50, based on the principle of residual connections. These connect the output of one layer to the input of another, deeper layer, to add input from lower layers to the upper layers of a network. In this way, the performance of deep networks is enhanced [24].
- EfficientNetB3, a model that effectively balances network width (number of filters or channels in convolutional layers), depth (number of layers) and resolution (size of inputs and outputs in convolutional layers) to optimize network performance [25].

All these networks have been trained using the ImageNet database, which currently contains over 14 million images, divided into 1,000 categories. The question then arises as to which model will be the most appropriate for classifying our dermatological images, and how many layers should be frozen.

At this point, it is essential to remember that the primary aim of this experiment with neural networks is not to design the most optimal model for detecting Monkeypox, but to study the impact that Maxwell could have on training these networks by filtering different datasets. A number of models have already been designed for the classification of Monkeypox snapshots, with remarkable performances, reaching 99.5% validation accuracy [26]. However, it would be pointless for us to directly adopt the model as it stands, with its weights already defined, without training it on our data, since we wouldn't be able to observe the effect of sub-sampling the training dataset. So we'll try to build an efficient model from scratch, drawing on existing ones, but we won't employ some of the more advanced and powerful optimization means, which would be far more time-consuming and resource-heavy. Furthermore, the methodology used to create the model will be generic, making it easy to adapt to other cases.

With this in mind, we build a new neural network, which includes the pre-trained model (with the exception of its last layer, intended for classification). Upstream of this is an image pre-processing layer, and downstream are a number of dense layers, enabling the model to adapt to the Monkeypox detection.

In this way, by using the various basic models mentioned above, and varying the number of layers we train, we can get an idea of the overall model best suited to our classification task by comparing the performance obtained. Network depth varies greatly from one basic model to another, so rather than making a fixed number of layers trainable, we'll freeze different proportions of total layers.

The results obtained will then take the following form: "For a global network, comprising a base model "A" with x% of layers trained, the classification yielded an accuracy of y%". We are looking for the combination (A, x) giving the highest y value.

Since the aim here is to highlight a trend, and since the measurements will be discrete, a regression is used. In this case, a degree 3 interpolation curve has been chosen; it allows us to estimate the values between the discrete data, while the degree guarantees an estimate faithful to the readings. This produces the following plots:

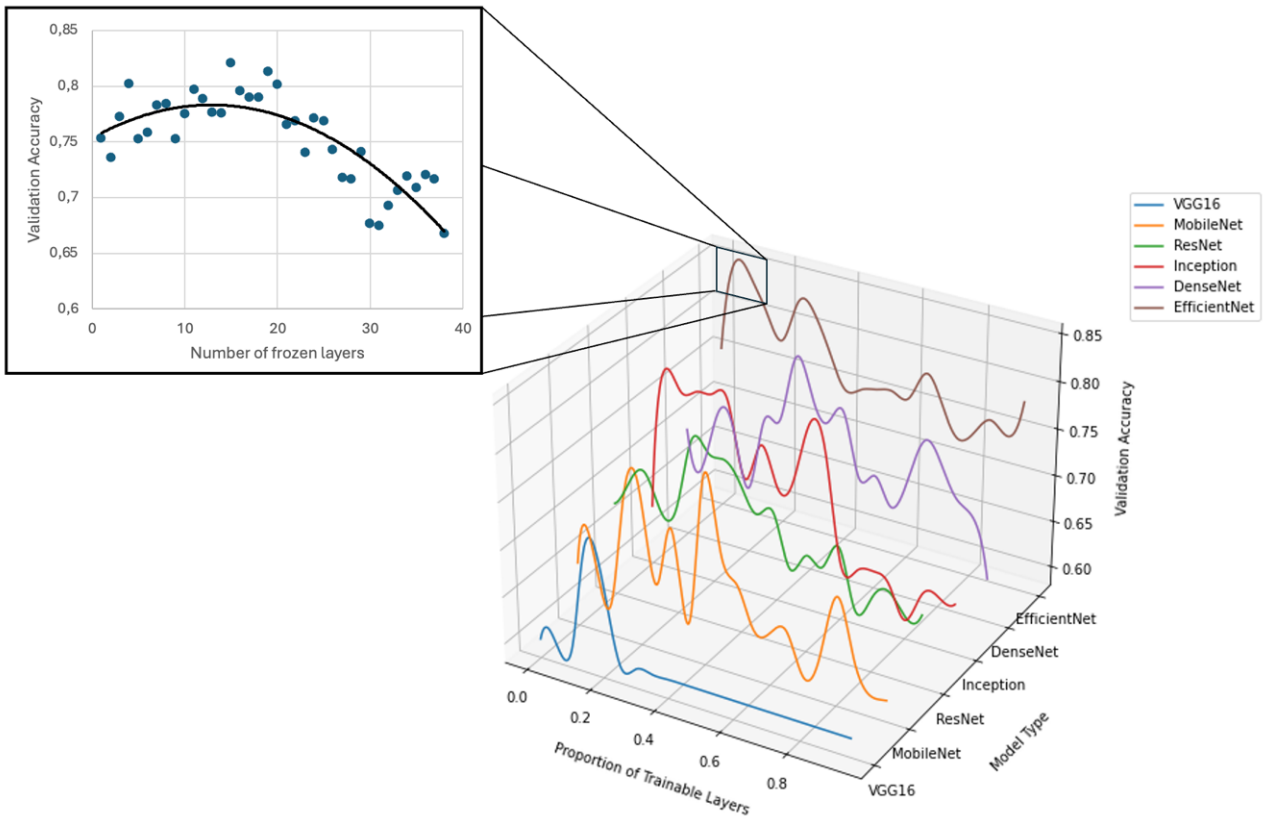


Figure 15: Benchmark of base models and proportions of trainable layers for Monkeypox classification

We can see that the EfficientNetB3 model stands out from the others, with higher accuracy values, particularly when training between 0 and 5% of the base model layers. Measurements over this reduced interval allow us to refine our choice to make only 14 layers trainable ($\sim 4\%$). This base model will therefore be retained for the rest of the project.

3.2.2.5 Light optimizations of the model

Despite the chosen architecture, the first results presented by the model are alarming: validation accuracy is well below training accuracy, and the validation loss function does not seem to converge. This again suggests overfitting. A first solution would be to perform data augmentation once again, but this would contradict the aim of the experiment, and would not necessarily be wise. Indeed, since the training dataset already includes augmented data, the model could end up overfitting the augmentations, rather than the original data. In this way, the specificities of the original data could become diluted, and harder for the model to capture.

Instead, another technique widely used in computer vision, the field of AI to which image classification belongs, is dropout. This consists in deactivating certain neurons during each train-

ing stage, thus preventing them from becoming too dependent on each other. In this way, the network learns more robust and generalizable features [27]. In practice, this technique is simply implemented via a Tensorflow function, keras.layers.Dropout, which randomly deactivates a certain percentage of neurons in a specified layer. Here, after testing several configurations, it was decided to perform a 30% dropout on the last included layer of the EfficientNetB3 model, followed by another 30% dropout on the next layer, the dense 256-neuron layer. This modification slightly improves the overfitting problem, but it still remains.

Moreover, it can sometimes be observed that the model's performance can deteriorate during the learning process: training accuracy increases with each epoch, but validation accuracy decreases, while the validation loss function increases. Rather than keeping the model as it was at the last epoch, it is possible to restore the parameters that gave the best performance (in this case, the validation accuracy is used as a comparator). This technique is called "callback". Customized callbacks can be defined, for example, using a metric other than validation accuracy as a reference, or stopping when the model reaches a certain performance threshold. Thus, for this implementation, restoring the model with the best validation performance should increase the model's ability to generalize its predictions.

Finally, some hyperparameter (number of epochs, number of neurons per layer, dropout rate, ...) optimization techniques could have been implemented to make the model even better. To this end, the GridSearchCV function, from the Sklearn Python package, was used. It searches for the optimal combination of hyperparameters to guarantee the best model performance. However, this is an extremely time-consuming technique, which did not deliver good reproducible results in this context.

Finally, the slightly optimized model will take the form shown in figure 16.

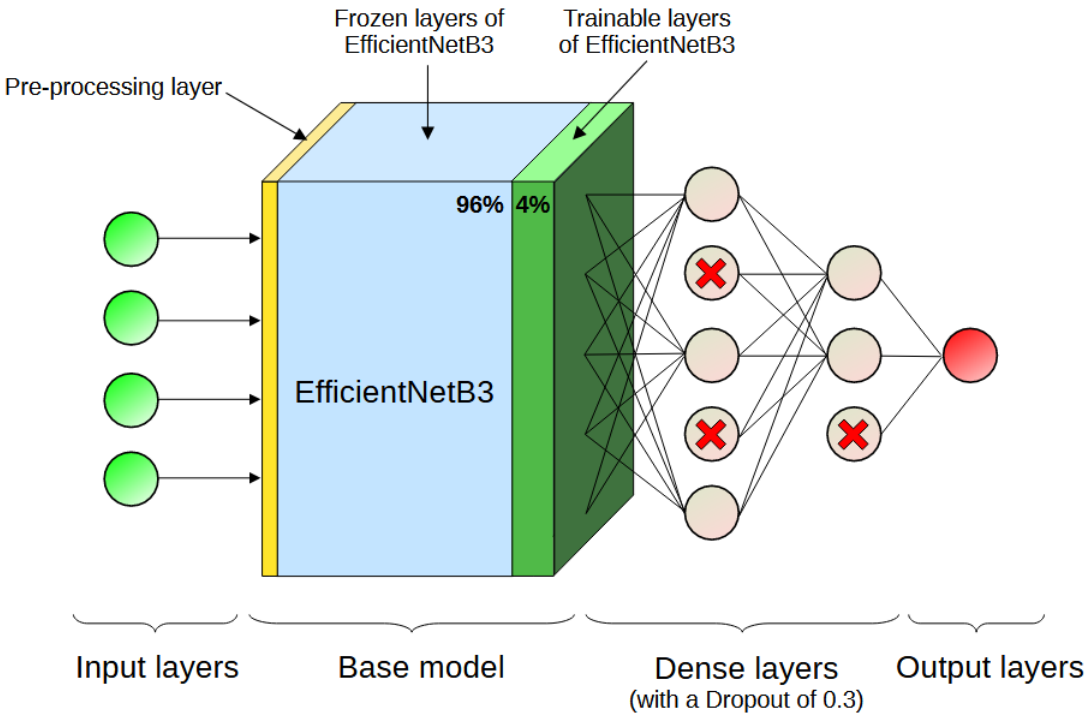


Figure 16: Architecture of the final model for Monkeypox classification

This model will be tested on the initial MSLD dataset in the following part.

3.2.2.6 Classification results of the model

Although model performance may differ slightly from one training session to the other, the paces of the training curves remain the same. An example is shown in figure 17.

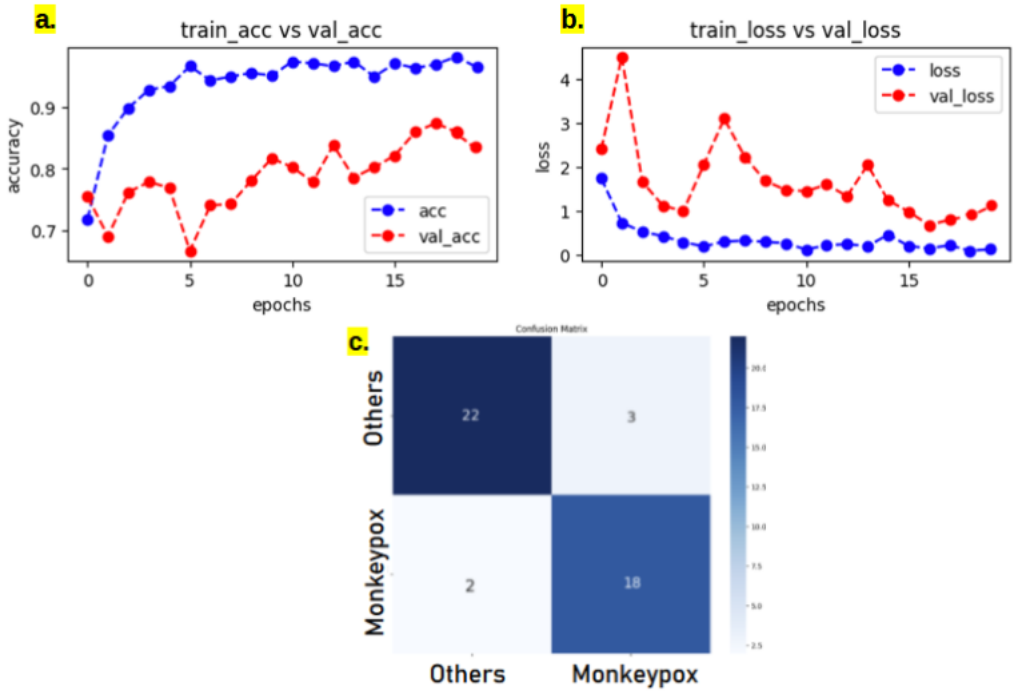


Figure 17: Examples of performance reached by the model

(a.) Evolution of the precision during the training. (b.) Evolution of the loss function during the training. (c.) Confusion matrix on the test dataset

In this example, we can see that a gap remains between validation and training performance. The model does not generalize its learning perfectly, and makes mistakes on data it has never seen. Nevertheless, it achieves a validation accuracy of around 88%, which is sufficient for further experimentation. Furthermore, we can see that the best performance is not achieved in the last training epoch, hence the usefulness of the callback mentioned earlier, which here restores the model weights of the antepenultimate epoch. Finally, the model is evaluated on the test set provided for this purpose by MSLD, and obtains the results entered in the confusion matrix, which reads as follows:

- Top left: the number of "Others" images predicted correctly. These are the true negatives.
- At top right are the "Others" images predicted as "Monkeypox", i.e. the false positives.
- Bottom left shows the number of "Monkeypox" images predicted as "Others", i.e. false negatives.
- At bottom right are the "Monkeypox" images correctly predicted, the true positives.

Finally, the PCA graph of the features extracted by the model has also been plotted (figure 18) to illustrate that our model is capable of extracting features that allow a better distinction between the two classes.

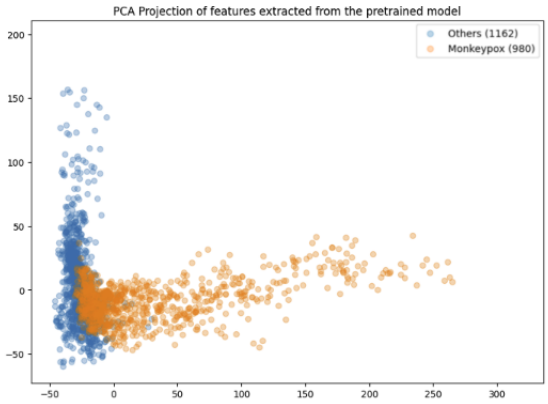


Figure 18: PCA graph of the features extracted by the model

Compared with the PCA drawn using the base data (figure 10), it is now easier to establish a boundary between the two classes, highlighting that the features extracted by the model are relevant for classification.

3.2.2.7 Modifications of datasets

As you may recall, the aim of this experiment is to analyze the performance of a model for different training bases deduced from Maxwell. It will therefore be necessary to extract sub-directories (barycenters, barycenter complements, monohypotheses and multihypotheses) from the initial dataset.

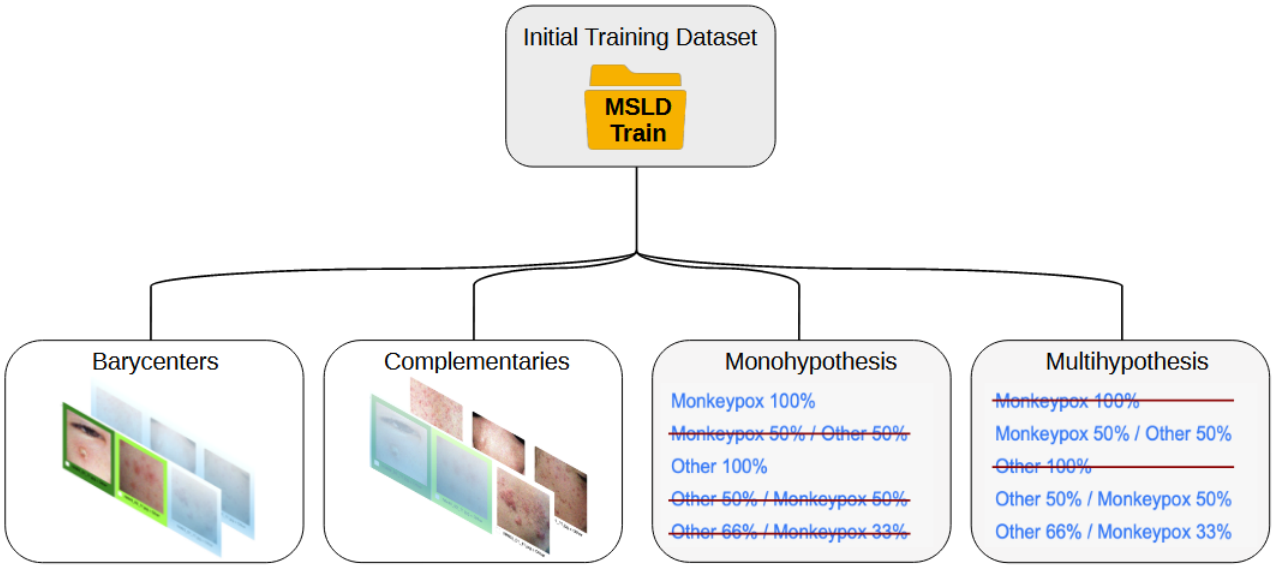


Figure 19: Graph of the different sub-sampled datasets

Finally, the last step is to structure the data correctly for integration into a neural network. Here, we build a Python dataframe which is a 2D structure, containing on the one hand the image pixel values, and on the other hand the associated label (0 for Other, and 1 for Monkeypox). Image pre-processing steps such as vector flattening and value normalization are carried out within the model.

3.2.2.8 Results of the hybridization and discussions

The model was therefore trained on 3 different data sets, and tested on 4 sets, depending on the training. To avoid excessive bias due to the initialization of weights when building the model, or variability of results that may differ due to dropouts, the experiment was repeated several times, before the results were averaged. In this case, the experiment was repeated a total of 13 times, corresponding to two days of uninterrupted calculation. The average performance of the model trained on the different bases deduced from Maxwell is shown in the following table:

Training dataset	Nb. of images	Training duration per epoch (s)	Test dataset	Nb. of tested images	Val_acc	Test_acc	Normalized confusion matrix	
MSLD Training set	2142	18	MSLD Test set	45	0.867	0.880	81,5%	18,5%
							3,8%	96,2%
Barycenters	652	8	MSLD Test set	45	0.848	0.865	81,5%	18,5%
							7,3%	92,7%
			Complementaries	1321	0.835	0.927	97,1%	2,9%
							12,1%	87,9%
			MSLD Test set + Complementaries	1366	0.844	0.937	96,9%	3,1%
							9,8%	90,2%
Mono-hypothesis	1952	16	MSLD Test set	45	0.878	0.862	78,8%	21,2%
							4,6%	95,4%
			Multi-hypothesis	11	0.867	0.986	100,0%	0,0%
							3,1%	96,9%

Table 1: Performances of the model trained on initial and sub-sampled datasets

From left to right: the training dataset used, the number of images it contains, the training duration at each epoch, the test dataset used, the number of images on which the model was tested, the validation accuracy, the test accuracy and finally the normalized confusion matrix.

First of all, the validation and test performances of the model trained on the MSLD Test set, equal to 0.867 and 0.880 respectively, will serve as a standard of comparison for this experiment. On the one hand, we note that by reducing the training dataset to barycenters, while keeping the same test set, performance decreases slightly. Indeed, we lose 1.9% in validation accuracy (0.867 $\rightarrow$ 0.848), and 1.5% in test accuracy (0.880 $\rightarrow$ 0.865). Nevertheless, the training time saved is considerable: from a duration per training epoch of 18 seconds for the initial training data set, to just 8 seconds for the reduced set. The question then arises as to whether this reduction in time is sufficient to justify a drop in performance, which will be addressed in the "Discussions and perspectives" section. Performance on barycenter complements and concatenation of the initial test base with complements, at 0.927 and 0.937 respectively, is high. This can be explained simply: a significant proportion of the complements are derived from transformations of the barycenters, which were present in the training set. The model is therefore evaluated on images close to those on which it was trained.

On the other hand, when the model is trained on the images Maxwell has grouped together as mono-hypotheses, and then evaluated on the initial test set, validation accuracy increases by 1.1% (0.867 $\rightarrow$ 0.878), while test accuracy decreases by 1.8% (0.880 $\rightarrow$ 0.862). However, even if validation is better, the main objective is to obtain better performance on data that has never been seen before. Moreover, while the reduction in test accuracy is greater in the first configuration, the time saving is less. In fact, by training only on mono-hypotheses, we save 2 seconds each time. With regard to the evaluation on multi-hypotheses, we can say that the accuracy of 0.986 does not necessarily denote an improvement in the model, but rather a low statistical power of the test, due to the small number of images (11) on which the model is evaluated.

Finally, we note that when the model is tested on the set initially intended for this purpose, it predicts far more false-positives than false-negatives. In the biomedical field, false-negatives are often more worrying than false-positives: a diagnosis that leads to further examinations proving negative later on, rather than a disease that is not detected in time, leading to delayed treatment

or even the spread of the disease. This is the basic principle of the diagnostic method. False positives and false negatives seem to be reversed when other test sets are used, but it must be borne in mind that in one case, the test set does not have the correct variability, and in the other, the statistical power is too low.

3.2.2.9 Discussions and prospects

The most striking result of this experiment is that Maxwell can save a considerable amount of time by training a model solely on the extracted barycenters, at the expense of a few percent of accuracy in the example studied. Obviously, the speed of a system does not systematically prevail over its accuracy, particularly in the biomedical field. Nevertheless, some applications choose to make this compromise, and it will then be necessary to see to what extent Maxwell can be implemented in the classification chain. This question is the subject of a separate approach in Machine Learning: approximate computing [28], which accepts partially correct results to optimize, among other factors, the efficiency or energy consumption of a system. However, this project is still in its infancy, and would require further experimentation in other fields of application, which could not be carried out before writing this report. For example, it would be necessary to test multi-class classification, or on training bases that have not been formed from data augmentation. These avenues will be investigated at a later date. In addition, it should be borne in mind that the model designed to be trained on the entire MSLD database is not necessarily the one that would perform best when trained on a sample, and would probably require some modifications to its architecture. These avenues will be investigated at a later date.

Conclusion

Although this internship was divided into two quite distinct missions, there was a coherence that I was looking for in responding to this offer. I was able to familiarize myself with artificial intelligence development by being confronted with two essential aspects of this sector: data retrieval and pre-processing, and model conception.

First of all, I was fortunate enough to be trained in some good programming practices. I was able to implement Test Driven Development (TDD) techniques, an approach that consists of systematically writing tests on every function created. I also got to grips with Git tools, enabling version management and project sharing. This learning process made me considerably more efficient throughout my project.

Secondly, the creation of the graphical interface and the Monkeypox experience taught me a lot about data preparation in general. On the one hand, I was confronted with "Big Data" challenges: large volumes of extremely varied data, for which particular care had to be taken in repatriation so as not to overload the servers. I also had to deal with the problem of dataset qualification. Indeed, in the first experiment, I was faced with incomplete, redundant and sometimes unverified data, which led me to question the consequences of such biases. While Maxwell overcomes these difficulties particularly well, this is not the case for other approaches such as neural networks, which are sensitive to these, and can see their performance greatly degraded. The second experiment, which involved qualifying a dataset, was particularly instructive in terms of the need to separate a dataset into several different sets in order to correctly evaluate a model. It also allowed me to study another aspect of data pre-processing, that of image normalization and resizing. In addition, my work on the Monkeypox was a perfect illustration of the advantages and disadvantages of data augmentation, a widely used technique. While the latter is often beneficial for improving the performance of a model, it can also turn out to be detrimental in some cases by introducing irrelevant noise, or redundancy in the data, preventing certain subtleties from being captured. Nevertheless, I didn't have the opportunity to tackle the issue of data labeling, although I had a sense of the complexity this task could represent through NCBI data.

On the other hand, this internship has enabled me to learn a great deal about the creation of classification tools. Firstly, I had the chance to discover the Maxwell platform, which is a novelty in this field thanks to the distance calculation on which it is based. Unlike neural networks, which take on the appearance of a black box, it is possible to understand Maxwell's entire pathway to results. By looking at how it works, I've been able to learn more about algorithmic information theory and gain an initial insight into common problems such as data compression and encoding. Secondly, experimenting on the Monkeypox enabled me to design my first neural networks by taking the TensorflowKeras library in hand. As well as gaining a wealth of knowledge about the architecture of different neural networks, I was able to devote some time to model optimization. Indeed, I learned how to use Transfer Learning, which is commonly used in classification problems, but also some particular regularization techniques such as the use of dropouts or callbacks. Even if some methods could have been developed further, such as the optimization of model hyperparameters, I was able to familiarize myself with many common practices, which I hope to be able to use again in the future.

Finally, during this internship, I was able to tackle a point echoing my previous internship in the CRMBM laboratory (Centre de Résonance Magnétique en Biologie et en Médecine), which is the processing of medical data. Indeed, certain constraints are inherent to this sector, such as the need for proof put forward by "5P" medicine, mentioned in the introduction. Over the past few months, I've been confronted with a very complex subject involving prognosis and diagnostic as-

sistance. However, this approach undermines our confidence in technology, since, while no system is infallible, it is crucial to know how to identify sources of error. I was particularly interested in this issue, especially as Maxwell offers a traceability solution that meets this need.

On top of this, during this internship, my work has already been put to good use: the interface I designed has been used to facilitate J. Demongeot's work, as well as to investigate new ways of classifying genomic data. As for the neural network programs, they form a solid basis for further experimentation, and will contribute to performance comparisons between Maxwell and Deep Learning approaches.

References

[1] Jumper J., Evans R., Pritzel A., et al., *Highly accurate protein structure prediction with AlphaFold*, Nature 596 (2021) 583-589.

[2] Newby D.E., Adamson P.D., Berry C., et al., *Coronary CT Angiography and 5-Year Risk of Myocardial Infarction*, The NEW ENGLAND JOURNAL of MEDICINE 379 (2018) 924-933.

[3] Mbuya-Bienge C., Kazemali C., Lapointe J., et al., *397 - Revue systématique des modèles de prédiction du cancer du sein avec un score polygénique*, Revue d'Épidémiologie et de Santé Publique (2022) 186.

[4] Debong F., Mayer H., Kober J., *Real-World Assessments of mySugr Mobile Health App*, Diabetes Technology & Therapeutics 21 (2019) 235-240.

[5] Liu C., Liu X., Wu F., et al., *Using Artificial Intelligence (Watson for Oncology) for Treatment Recommendations Amongst Chinese Patients with Lung Cancer: Feasibility Study*, J Med Internet Res. 20(9) (2018).

[6] Gardes J., *Le document numérique : la complexité des formes et les formes de la complexité*, (2009) 125-140.

[7] Cassidy B., Kendrick C., Brodzicki A., et al., *Analysis of the ISIC image datasets: Usage, benchmarks and recommendations*, Medical Image Analysis 75 (2022).

[8] Tsoukalas A., Albertson T., Tagkopoulos I., *From data to optimal decision making: a data-driven, probabilistic machine learning approach to decision support for patients with sepsis*, JMIR Med Inform. 3(1) (2015).

[9] du Plessis S. J., Blaxter M., Chadwick E. A., et al., *Genomics Reveals Complex Population History and Unexpected Diversity of Eurasian Otters (Lutra lutra) in Britain Relative to Genetic Methods*, Mol Biol Evol 40(11) (2023).

[10] Talibzade R., *Sentiment Analysis of IMDb Movie Reviews Using Traditional Machine Learning Techniques and Transformers*, (2023).

[11] Sayers E., *Sample Applications of the E-utilities*, Entrez Programming Utilities Help [Internet] (2010).

[12] Gardes J., Maldivi C., Boisset D., et al., *An Unspervised Classifier for Whole-Genome Phylogenies, the Maxwell © Tool*, International Journal of Molecular Sciences 24(22) (2023).

[13] Petitjean C., Deschamps P., López-García P., et al., *Extending the conserved phylogenetic core of archaea disentangles the evolution of the third domain of life*, Mol. Biol. Evol. 32 (2015) 1242-1254.

[14] Sokhna M., *Mise en œuvre d'un réseau de neurones sur un corpus d'apprentissage issu de la plateforme Maxwell*, (2022).

[15] McCulloch W.S., Pitts W., *A logical calculus of the ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics 5 (1943) 115–133.

[16] Ng A., *Neural Networks Overview*, Deeplearning.ai.

[17] Ali S. N., Ahmed M. T., Paul J., et al., *Monkeypox Skin Lesion Detection Using Deep Learning Models: A Preliminary Feasibility Study*, arXiv (2022).

[18] Ali S. N., Ahmed M. T., Jahan T., et al., *A Web-based Mpox Skin Lesion Detection System Using State-of-the-art Deep Learning Models Considering Racial Diversity*, arXiv (2023).

[19] Zhang M., Dong C., Fu J., et al., *AdaSelection: Accelerating Deep Learning Training through Data Subsampling*, arXiv (2023).

[20] Simonyan K., Zisserman A., *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv (2015).

[21] Sandler M., Howard A., Zhu M., et al., *MobileNetV2: Inverted Residuals and Linear Bottlenecks*, arXiv (2019).

[22] Huang G., Liu Z., van der Maaten L., et al., *Densely Connected Convolutional Networks*, arXiv (2018).

[23] Szegedy C., Vanhoucke V., Ioffe S., et al., *Rethinking the Inception Architecture for Computer Vision*, arXiv (2015).

[24] He K., Zhang X., Ren S., et al., *Deep Residual Learning for Image Recognition*, arXiv (2015).

[25] Tan M., Le Q.V., *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*, arXiv (2020).

[26] Nayak T., Chadaga K., Sampathila N., et al., *Deep learning based detection of monkeypox virus using skin lesion images*, Medicine in Novel Technology and Devices 18 (2023).

[27] Ng A., *Regularizing your neural network*, Deeplearning.ai.

[28] Moreau T.; San Miguel J.; Wyse M., et al., *A Taxonomy of General Purpose Approximate Computing Techniques*, IEEE Embedded Systems Letters 10 (1) (2018) 2-5.

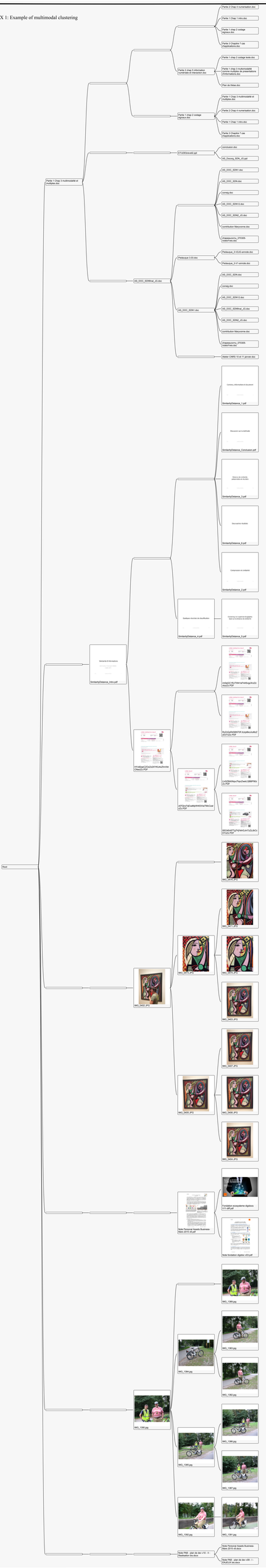
Summary

This internship was divided into two distinct missions. Firstly, to understand the issues involved, it was necessary to be familiar with the Maxwell platform, a tool for the agnostic, unsupervised clustering of multimodal digital content. Once the concepts of algorithmic information theory underlying Maxwell had been acquired, the first mission could begin. This consisted in improving the repatriation of genomic data from NCBI and then classifying them with Maxwell. The source of the data raised a number of problems: the database is imprecise and incomplete, and requires particular attention to be paid to the volume of data repatriated. To meet the initial need, a graphical interface was designed to download and pre-process the genomes of numerous species to make them usable by Maxwell. In addition, a number of complementary tools were added to the interface to make the best use of a second approach, AL distance, to verify the genomic classifications obtained by Maxwell. The second mission was in response to a request for the development of an application to enhance the detection of monkeypox, an emerging disease in Africa. The aim here was to qualify datasets by comparing the performance of a neural network trained on a complete database, then on sub-sampled databases deduced from Maxwell clustering. Through this experiment, several fundamental concepts of Deep Learning model design, particularly Transfer Learning, were addressed.

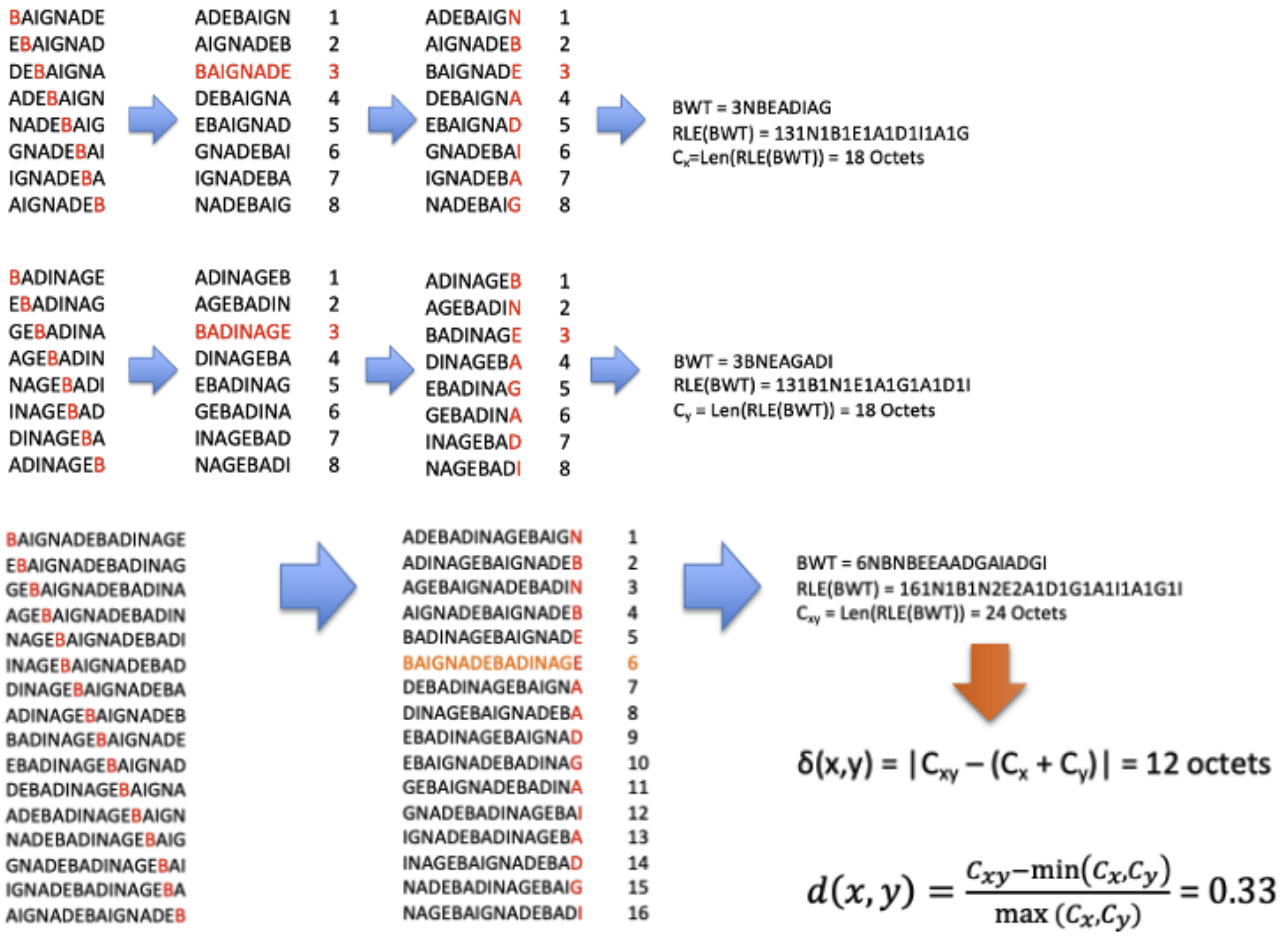
Résumé

Ce stage a été divisé en deux missions distinctes. Tout d'abord, pour comprendre les enjeux relatifs à celles-ci, il était nécessaire d'être familier avec la plateforme Maxwell, un outil permettant le clustering de contenus numériques multimodaux, de manière agnostique et non-supervisée. Une fois les concepts de la théorie algorithmique de l'information sous-jacents à Maxwell acquis, la première mission pouvait débuter. Celle-ci consistait à améliorer le rapatriement des données génomiques depuis NCBI pour ensuite les classifier avec Maxwell. La provenance des données soulevait plusieurs problèmes : la base de données est imprécise, lacunaire, et nécessite une attention particulière aux volumes de données rapatriés. Pour répondre au besoin de départ en intégrant ces contraintes, une interface graphique a été conçue, permettant de télécharger puis pré-traiter les génomes de nombreuses espèces pour les rendre exploitables par Maxwell. De plus, certains outils complémentaires ont été ajoutés à l'interface pour utiliser au mieux une seconde approche, la distance AL, permettant de vérifier les classifications génomiques obtenues par Maxwell. La seconde mission répondait à une demande concernant le développement d'une application renforçant la détection de la variole du singe, une maladie émergente en Afrique. Le but était ici de qualifier des jeux de données en comparant les performances d'un réseau de neurones entraînés sur une base complète, puis sur des bases sous-échantillonnées déduites du clustering de Maxwell. A travers cette expérimentation, plusieurs concepts fondamentaux de la conception de modèle de Deep Learning, tout particulièrement le Transfer Learning, ont été abordés.

APPENDIX 1: Example of multimodal clustering



APPENDIX 2: Example of distance computation between words "baignade" and "badinage"



APPENDIX 3: Complementaries documents produced by Maxwell

sequence length = 1664970

1) AAGAT count = 4721
2) AGATG count = 3306
3) ATGAA count = 3893
4) ATTCA count = 2412
5) CAAGA count = 1563
6) GATGA count = 3135
7) TCAAG count = 925
8) TGAAT count = 2460
9) TTCAA count = 4230

all) sum = 26645

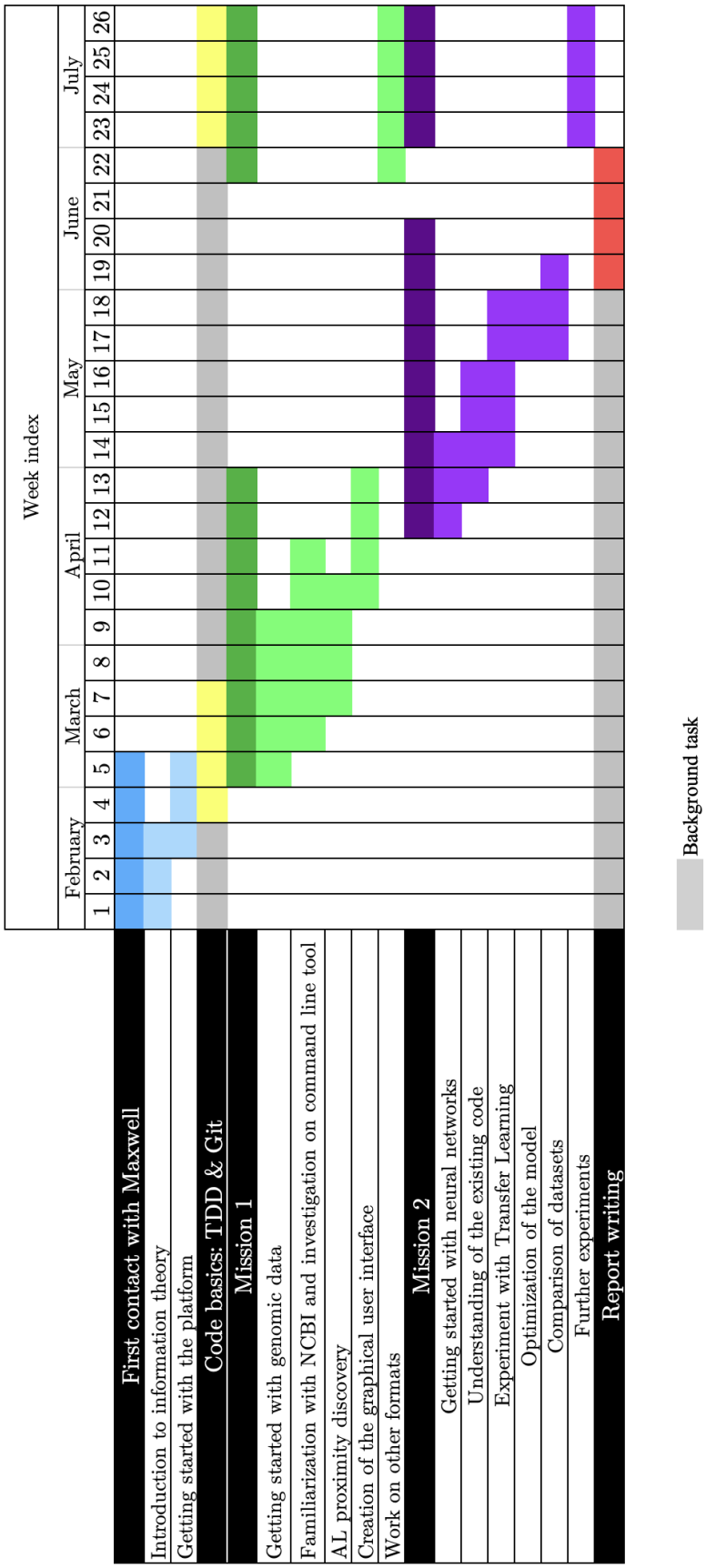
Significance = 99.73 (n=1664966; A=14633.49; std=120.44)

Example of HTML file generated for a *methanococci* sequence

bio_parser - 1.4.0 - 08/04/2024 09:44:06	Significance	AAGAT	AGATG	ATGAA	ATTCA	CAAGA	GATGA	TCAAG	TGAAT	TTCAA	Total
NC_013156_1_Methanocaldococcus_fervens_AG86__complete_sequence	Significance = 95.54(n=1485057; A=13052.26; std=113.74)	4020	2831	3532	2300	1404	2693	822	2157	4160	23919
NC_013157_1_Methanocaldococcus_fervens_AG86_plasmid_pMEFER01__complete_sequence	Significance = 6.83(n=22186; A=194.99; std=13.9)	40	24	28	26	22	27	19	38	66	290
NC_009635_1_Methanococcus_aeolicus_Nankai3__complete_sequence	Significance = 70.52(n=1569496; A=13794.4; std=116.93)	2907	2006	3793	2784	1213	1544	905	2888	4000	22040
NZ_CP009149_1_Methanocaldococcus_bathoardescens_strain_JH146_chromosome__complete_genome	Significance = 102.81(n=1607552; A=14128.88; std=118.34)	4653	3140	3943	2448	1615	2832	867	2404	4393	26295
NC_009634_1_Methanococcus_vannielii_SB__complete_sequence	Significance = 119.89(n=1720044; A=15117.57; std=122.41)	3270	1829	4372	3545	1911	1819	2120	3536	7391	29793
NC_015562_1_Methanotorris_igneus_Kol_5__complete_sequence	Significance = 112.75(n=1854193; A=16296.62; std=127.1)	4882	3311	4810	3027	2025	3496	1160	3033	4883	30627
NZ_CP026606_1_Methanococcus_maripaludis_strain_DSM_2067_chromosome__complete_genome	Significance = 121.76(n=1714914; A=15072.49; std=122.23)	3548	1843	4161	3518	1974	2069	2100	3746	6996	29955
NC_015636_1_Methanothermococcus_okinawensis_IH1__complete_sequence	Significance = 80.16(n=1662521; A=14612.0; std=120.35)	3361	2222	3984	2904	1428	2308	1046	2900	4106	24259
NC_015632_1_Methanothermococcus_okinawensis_IH1_plasmid_pMETOK01__complete_sequence	Significance = 13.93(n=14926; A=131.19; std=11.4)	50	36	57	25	14	37	6	28	37	290
NZ_LR792632_1_Methanocaldococcus_lauensis_strain_SG7_chromosome_I__complete_sequence	Significance = 96.04(n=1532281; A=13467.31; std=115.54)	4560	2936	3786	2371	1446	2480	702	2352	3931	24564
NZ_LR792633_1_Methanocaldococcus_lauensis_strain_SG7_plasmid_pI__complete_sequence	Significance = 4.61(n=5138; A=45.16; std=6.69)	18	12	9	5	4	10	2	6	10	76
NC_013407_1_Methanocaldococcus_vulcanius_M7__complete_sequence	Significance = 94.62(n=1746325; A=15348.56; std=123.34)	5353	3199	3728	2343	1870	2813	1042	2511	4160	27019
NC_013408_1_Methanocaldococcus_vulcanius_M7_plasmid_pMETVU01__complete_sequence	Significance = 0.31(n=10700; A=94.04; std=9.65)	19	11	17	3	18	12	6	4	7	97
NC_013409_1_Methanocaldococcus_vulcanius_M7_plasmid_pMETVU02__complete_sequence	Significance = 2.14(n=4700; A=41.31; std=6.4)	6	2	6	13	1	5	2	4	16	55
NC_014122_1_Methanocaldococcus_infernus_ME__complete_sequence	Significance = 77.18(n=1328190; A=11673.54; std=107.57)	4593	2709	2610	1334	1425	2076	1071	1401	2757	19976
NC_000909_1_Methanocaldococcus_jannaschii_DSM_2661__complete_sequence	Significance = 99.73(n=1664966; A=14633.49; std=120.44)	4721	3306	3893	2412	1563	3135	925	2460	4230	26645
NC_001732_1_Methanocaldococcus_jannaschii_DSM_2661_plasmid_pDSM2661_1__complete_sequence	Significance = 17.32(n=58403; A=513.31; std=22.56)	122	52	117	112	40	68	51	118	224	904
NC_001733_1_Methanocaldococcus_jannaschii_DSM_2661_plasmid_pDSM2661_2__complete_sequence	Significance = 4.96(n=16546; A=145.42; std=12.01)	28	10	22	29	7	10	11	22	66	205
NC_013156_1_Methanocaldococcus_fervens_AG86__complete_sequence	Significance = 95.54(n=1485057; A=13052.26; std=113.74)	4020	2831	3532	2300	1404	2693	822	2157	4160	23919
NC_013157_1_Methanocaldococcus_fervens_AG86_plasmid_pMEFER01__complete_sequence	Significance = 6.83(n=22186; A=194.99; std=13.9)	40	24	28	26	22	27	19	38	66	290
NC_009635_1_Methanococcus_aeolicus_Nankai3__complete_sequence	Significance = 70.52(n=1569496; A=13794.4; std=116.93)	2907	2006	3793	2784	1213	1544	905	2888	4000	22040
NZ_CP009149_1_Methanocaldococcus_bathoardescens_strain_JH146_chromosome__complete_genome	Significance = 102.81(n=1607552; A=14128.88; std=118.34)	4653	3140	3943	2448	1615	2832	867	2404	4393	26295
NC_009634_1_Methanococcus_vannielii_SB__complete_sequence	Significance = 119.89(n=1720044; A=15117.57; std=122.41)	3270	1829	4372	3545	1911	1819	2120	3536	7391	29793
NC_015562_1_Methanotorris_igneus_Kol_5__complete_sequence	Significance = 112.75(n=1854193; A=16296.62; std=127.1)	4882	3311	4810	3027	2025	3496	1160	3033	4883	30627
NZ_CP026606_1_Methanococcus_maripaludis_strain_DSM_2067_chromosome__complete_genome	Significance = 121.76(n=1714914; A=15072.49; std=122.23)	3548	1843	4161	3518	1974	2069	2100	3746	6996	29955
NC_015636_1_Methanothermococcus_okinawensis_IH1__complete_sequence	Significance = 80.16(n=1662521; A=14612.0; std=120.35)	3361	2222	3984	2904	1428	2308	1046	2900	4106	24259
NC_015632_1_Methanothermococcus_okinawensis_IH1_plasmid_pMETOK01__complete_sequence	Significance = 13.93(n=14926; A=131.19; std=11.4)	50	36	57	25	14	37	6	28	37	290
NZ_LR792632_1_Methanocaldococcus_lauensis_strain_SG7_chromosome_I__complete_sequence	Significance = 96.04(n=1532281; A=13467.31; std=115.54)	4560	2936	3786	2371	1446	2480	702	2352	3931	24564
NZ_LR792633_1_Methanocaldococcus_lauensis_strain_SG7_plasmid_pI__complete_sequence	Significance = 4.61(n=5138; A=45.16; std=6.69)	18	12	9	5	4	10	2	6	10	76
NC_013407_1_Methanocaldococcus_vulcanius_M7__complete_sequence	Significance = 94.62(n=1746325; A=15348.56; std=123.34)	5353	3199	3728	2343	1870	2813	1042	2511	4160	27019
NC_013408_1_Methanocaldococcus_vulcanius_M7_plasmid_pMETVU01__complete_sequence	Significance = 0.31(n=10700; A=94.04; std=9.65)	19	11	17	3	18	12	6	4	7	97
NC_013409_1_Methanocaldococcus_vulcanius_M7_plasmid_pMETVU02__complete_sequence	Significance = 2.14(n=4700; A=41.31; std=6.4)	6	2	6	13	1	5	2	4	16	55

Example of CSV summary for the taxon *methanococci*

APPENDIX 4: Gantt diagram of completed work



APPENDIX 5: Archive sheet

- Identity: Clément BARAILLE (42101650)
- Branch: Biomedical
- Academic year: 2023-2024
- Internship headline and dates: "Stage - Apprentissage artificiel classification et IA - F/H" (05/02/2024-30/07/2024)
- Logo, name and postal address of company/laboratory: ORANGE LABS, 28 Chemin du Vieux Chêne, 38240, MEYLAN, FRANCE

- First name, last name and e-mail address of internship supervisor: Joël Gardes (joel.gardes@orange.com)
- First name, last name of school tutor: Alice Caplier
- The internship description:

  "Participate in the Maxwell research project (e-Health), an Orange Labs research project in collaboration with the Grenoble Faculty of Medicine (AGEIS laboratory, Pr. J. Demongeot), which has two complementary components: i) development of a classifier for processing complex biomedical data of all kinds (paraclinical and biological), ii) specific application in genetics and proteomics, which aims to innovate in terms of classification performance and ease of a posteriori interpretation of clusters.
  The work will be carried out along two complementary axes:
  1) Work on data corpora (acquisition, storage and scheduling)The data corpora belong to two distinct families: - Genomic data from the NCBI/NIH site (Bethesda, USA), covering the genomes of around 5% of the 9 million different species known on earth, accessible online ("The World's largest repository of medical and scientific abstracts, full-text articles, books and reports"). NCBI's GenBank® (www.ncbi.nlm.nih.gov/genbank/), for example, is a public database containing over 6.25 trillion DNA base pairs from more than 1.6 billion nucleotide sequences, for 450,000 formally described species.
  2) Work on data classification (clustering and interpretation of phylogenetic trees)The work has two main focuses:
  - The study of the origin of living organisms, through the construction of phylogenetic trees based on the Maxwell® classifier algorithm using the Burrows-Wheeler compression method and the associated Vitányi distance, then their interpretation in phylogenetic trees and those obtained using the AL-pentameric distance to primitive RNAs.
  - Comparison of classifications obtained from nuclear genomes (Maxwell® or classical origin) and mitochondrial genomes, which are much smaller in nucleotide number (about 2,105 bp content).

- Resources and supervisory staff: Development in Python, within a team containing both researchers and engineers.