University of Manouba
National School of Computer Science

Summer Internship Report

LabMentor

Prepared by :
***Riadh Belgacem**

Organization : Talan Tunisia
Supervised by : Houssem Ben Aicha
Address : Charguia 1, 2035 Tunisia
TEL : 70 01 50 06 *FAX* :70 01 50 07
eMail : ons.mahsni@talan.com

Academic Year :
2023-2024

Table des matières

General Introduction

This report presents the LabMentor project, developed by the "Nighty Nine" team during Talan SummerCamp 2024, as part of the theme "Technological Innovation as a Lever for Scientific Research." LabMentor is an innovative solution that combines artificial intelligence, mixed reality, and quantum computing to enhance the precision, safety, and efficiency of laboratory experiments.

Talan, a company committed to technological innovation, identified the need to strengthen the management of experimental procedures and ensure the reproducibility of results. LabMentor addresses this issue by offering a solution that optimizes laboratory processes while minimizing risks and ensuring compliance with regulations.

The artificial intelligence model used, based on Generative Adversarial Networks (GANs), is particularly powerful for generating high-resolution cellular images, which are essential for biological analysis. Additionally, a quantum computing algorithm has been implemented to analyze DNA sequences by encoding them into quantum superposition states and measuring their similarity.

This report is structured into three parts : the presentation of the context and challenges, the description of the technologies and proposed solutions, and a conclusion addressing future perspectives.

Chapitre 1

Company Discovery

1.1 Company Presentation

— Talan is an international consulting group specializing in transformation through Innovation, Technology, and Data. Founded in 2002 by Mehdi Houas, Eric Benamou, and Philippe Cassoulat, Talan advises and supports businesses and public institutions in implementing their transformation and innovation projects in France and internationally. Present in 18 countries, the group, certified Great Place To Work, has over 5000 employees and aims to achieve a turnover of 630 million euros in 2024, with a goal of exceeding one billion euros by 2026.

— In Tunisia and internationally, Talan specializes in new information and communication technologies (ICT), with a focus on customer relations. The group's sector expertise covers various fields such as financial services, insurance, telecommunications and media, energy, public services, as well as transportation and logistics.

— Alongside a Research and Innovation Center, Talan places innovation at the heart of its development and operates in key areas such as Artificial Intelligence, Data Intelligence, and Blockchain. With its business, functional, and technological expertise, Talan supports the growth of large groups and mid-sized companies with a committed and responsible approach.

1.2 External Diagnosis

1.2.1 Opportunities

— **International Presence** : Talan's international expansion allows the company to access diverse markets, attract a global clientele, and adapt to local requirements, thereby enhancing its competitiveness and brand recognition.
— **Growth in Key Sectors** : Talan is well-positioned to meet the growing demand, particularly in finance, where institutions seek innovative digital solutions and increased compliance.

1.2.2 Threats

— **Intense Competition** : With globalization, Talan faces increased competition from international firms and startups offering similar services at competitive prices, which threatens its market share.

1.3 Internal Diagnosis

1.3.1 Company Strengths

— **Innovation Capability** : Talan stands out for its ability to integrate the latest technologies and develop innovative solutions, meeting the changing needs of its clients and differentiating itself in a competitive market.
— **Qualified Team** : Talan's employees, with their solid technical and functional expertise, ensure high-quality services, enhancing client trust and satisfaction.

1.3.2 Company Weaknesses

— **High Workload** : Engineers at Talan face a heavy workload and long hours, which can lead to burnout, affecting work quality and employee satisfaction.

1.4 Proposed IT Solution

1.4.1 Problem Statement

In biology laboratories, researchers face complex challenges related to in vivo (on living organisms) and in vitro (outside the organism, in controlled conditions) approaches. While these methods are essential, they present limitations in terms of cost, time, and reproducibility. In silico simulations (computer modeling) offer an innovative alternative. They allow for optimizing experimental protocols, accelerating scientific progress, and reducing risks associated with handling hazardous substances. By integrating in silico methods into research processes, it becomes possible to control costs while meeting strict regulatory requirements.

1.4.2 Solution Description

To address the challenges encountered in biology laboratories, we propose **LabMentor**, an intelligent assistant incorporating mixed reality, artificial intelligence (AI), and Quantum Computing technologies.

1.4.3 Technological Choices

The tools and frameworks used include **Unity 3D** for creating the virtual environment compatible with HoloLens, **MRTK (Mixed Reality Toolkit)** for mixed reality development, and a

FIGURE 1.1 – LabMentor

REST API for managing experimental data in real time. These technological choices ensure robustness and scalability, allowing **LabMentor** to adapt to new features as it develops. "'

Chapitre 2

Specification of Requirements

2.1 Specification and Analysis of Requirements

2.1.1 Prediction of Fluorescent Nuclei Images

The first task was to predict fluorescent images that highlight cell nuclei from transmitted light images. This approach, already proven feasible, served as a starting point to explore different configurations of the *In Silico Labeling* (ISL) method and to become familiar with the data. For this task, we worked with two datasets : one using brightfield and phase-contrast images to predict DAPI images, and another using DIC images to predict the Hoechst channel. In both cases, the cell nuclei are labeled with fluorescence.

2.1.2 *Pattern Matching* Algorithm in Quantum Computing

Pattern Matching is one of the fundamental algorithms in computer science, with significant potential benefits through quantum computers. These algorithms are commonly used in image processing, DNA sequence analysis, as well as in data compression and statistics. Speeding up *Pattern Matching* using a quantum computer, particularly through Grover's search algorithm, would be a major advancement in all these fields.

Chapitre 3

Fluorescence images prediction

3.1 State of the Art

Recent developments in neural networks and deep learning approaches have greatly improved the performance of many computer vision tasks. Throughout this section, we present how Convolutional neural networks have become a groundbreaking tool for many tasks such as image segmentation. We also introduce Generative Adversarial Networks and develop their exciting adversarial framework.

3.1.1 Convolutional neural networks (CNN)

In this section, we define Convolutional neural networks (CNN), a class of artificial neural networks that has become dominant in various computer vision tasks and has attracted interest across many domains. CNN allow to extract essential features from images and to solve specific tasks by using multiple building blocks such as convolution layers, pooling layers and fully connected layers.

3.1.1.1 Feature Extraction : Convolution Block

A Convolutional Neural Network (CNN) uses convolution blocks to extract features from input images. These blocks include convolutional layers and pooling layers.

Filters (or kernels) are small matrices applied to an input image to detect specific features. Each filter performs a convolution operation, where it multiplies element-wise with a patch of the image and sums the result to produce a single value. This process is repeated across the image, creating a "feature map" that highlights the detected features.

Filters may be applied with different strides, allowing them to detect features at various positions in the image. Once a feature map is generated, it often passes through a nonlinear function like ReLU, similar to fully connected layers.

3.1.1.2 CNN Architecture

CNNs consist of multiple convolutional layers. During training, the network learns filter weights through backpropagation to minimize task-specific loss. For instance, in a task like classifying dogs vs. cats, filters might learn to identify features like ear shapes.

CNNs typically use multiple filters to extract diverse features from the input. In multi-channel images (e.g., RGB), filters match the image's depth. For an image with L channels and K desired feature maps, K filters of size $(N \times M \times L)$ are used, resulting in $(N \times M \times L+1) \times K$ parameters, including biases.

3.1.1.3 Hierarchical Feature Learning

CNNs learn hierarchical features through successive layers. Early layers detect basic features like edges, while deeper layers identify complex patterns or objects..

Each unit in a convolutional layer is connected to a local region of the previous layer's feature map, significantly reducing the number of parameters compared to fully connected layers. This localized connectivity allows CNNs to efficiently handle high-dimensional inputs, such as images, by focusing on specific subsets of the image. The "receptive field" of a unit describes the portion of the input image it processes, with deeper layers covering larger areas.

3.1.2 Stride, Padding, and Pooling Layers

3.1.2.1 Stride

In a convolutional layer, the stride parameter controls the filter's movement across the image. For example, a stride of (1, 1) moves the filter one pixel at a time horizontally and vertically, while a stride of (2, 2) moves it two pixels at a time. The stride affects the size of the resulting feature map. For instance, convolving a 3 × 3 image with a 2 × 2 filter and a stride of (1, 1) produces a 2 × 2 feature map.

3.1.2.2 Padding

Padding adjusts the size of the feature map. "Valid padding" drops parts of the image where the filter doesn't fit, resulting in a smaller feature map. "Same padding" adds zeros around the image so that the output size matches the input size. This ensures that the filter fits the entire image.

3.1.2.3 Pooling

Pooling layers reduce the spatial size of feature maps, retaining essential information while minimizing dimensionality. This helps lower the number of parameters and computation required. Pooling is typically applied after the convolution and activation functions. It can be Max Pooling, which extracts the maximum value from each patch, or Average Pooling, which

calculates the average value. For example, a 2 × 2 pooling filter with a stride of (2, 2) halves the spatial dimensions of the feature map, turning a 6 × 6 map into a 3 × 3 map.

3.1.3 Image Segmentation and U-Net Model

Image segmentation is a technique used to divide an image into segments for easier analysis. U-Net is a widely-used architecture for semantic segmentation, characterized by its symmetric "U" shape.

U-Net consists of two main parts :

— **Contracting Path (Encoder) :** This path applies repeated 3x3 convolutions with ReLU activations and 2x2 max pooling with stride 2 for downsampling. The number of feature channels doubles after each downsampling step.
— **Expansive Path (Decoder) :** This path involves upsampling the feature maps using 2x2 "up-convolutions" to halve the number of feature channels. It then concatenates with the corresponding feature maps from the contracting path and applies two 3x3 convolutions. Cropping is used to handle the loss of border pixels during convolutions.

At the final layer, a 1x1 convolution maps the feature vectors to the desired number of classes. The network has a total of 23 convolutional layers, effectively balancing feature extraction and spatial resolution recovery.

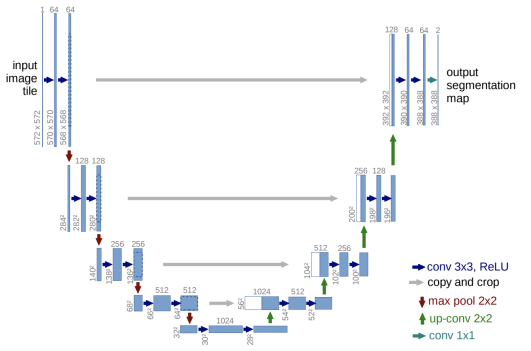

Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

FIGURE 3.1 – U-Net architecture

3.1.4 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) are a type of deep learning model designed for generating realistic samples. GANs consist of two neural networks, the generator and the discriminator, which are trained simultaneously with opposing goals.

— **Generator (G) :** The generator creates samples that aim to mimic the real data distribution. It generates "fake" samples from random noise.

— **Discriminator (D) :** The discriminator evaluates whether a given sample is real (from the training data) or fake (generated by the generator).

During training, the generator improves its ability to produce realistic samples to "fool" the discriminator, while the discriminator enhances its ability to distinguish between real and fake samples. This adversarial process helps the generator create high-quality, high-resolution images.

Mathematically, GANs are framed as a minimax game :

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

where G is the generator, D is the discriminator, and p_{data} is the real data distribution.

The game reaches a Nash equilibrium when :

$$p_{G^*} = p_{data} and D^*(x) = \frac{1}{2}$$

However, achieving this equilibrium can be challenging in practice.

GANs are trained using backpropagation with alternating gradient descent for the generator and discriminator, refining each network's performance through iterative updates.

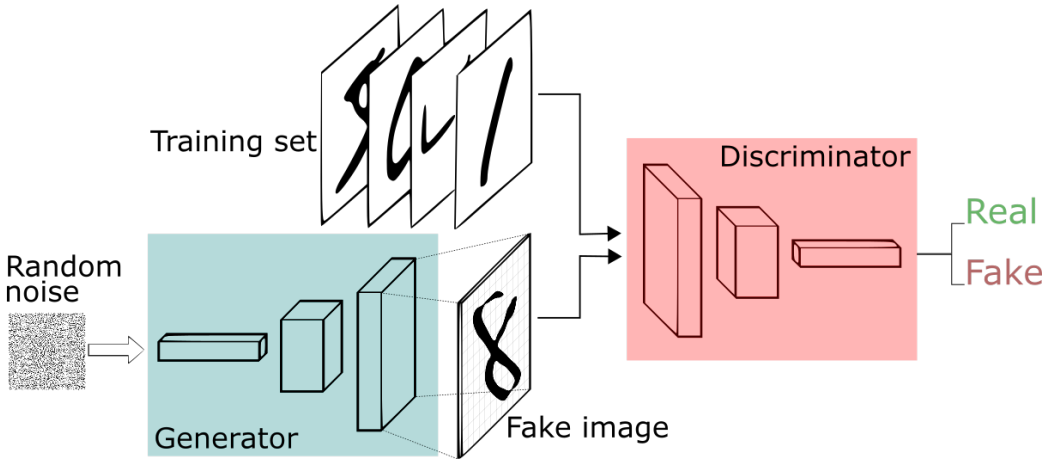

FIGURE 3.2 – GAN architecture

3.2 Data and Preprocessing of Keratinocytes (KC)

The keratinocyte (KC) data were collected at 60 minutes per image using standard DAPI, CY5, and YFP filters. DAPI is used to highlight the cell nucleus where the DNA is located. The objective is to develop an *in silico* labeling model capable of predicting DAPI fluorescent images from transmitted light images (Brightfield and phase contrast). The data are divided into training, testing, and validation sets, with each image being normalized using Z-score normalization :

$$newpixel = \frac{pixel - \mu}{\sigma},$$

where μ is the mean and σ is the standard deviation of the image. Due to memory limitations, the models are trained on 256x256 pixel patches. Data augmentation is performed by applying flips, rotations, and distortions, increasing the number of images from 1000 to 5000.

3.3 Model Benchmarking

To tackle our *in silico* labeling task, we chose to apply a U-Net architecture. We compared the performance of a classical U-Net with a pre-trained U-Net (referred to as U-Net on steroids) with deeper layers. Our models were trained using the different loss functions discussed in the previous section, and their performance was evaluated accordingly.

Initially, we began benchmarking with a Generative Adversarial Network (GAN) model. However, due to time constraints, we were unable to complete the training and evaluation of the GAN model. Therefore, our primary focus in this section remains on the U-Net architectures and their performance metrics.

By comparing the classical U-Net and the U-Net on steroids, we aim to assess which configuration provides better results for the task at hand and understand the implications of using pre-trained models versus training from scratch. This comparison is critical in identifying the most effective approach for our *in silico* labeling task and addressing the challenges we faced during experimentation.

3.4 Model Evaluation

To evaluate our models, we use the Pearson correlation coefficient (PCC), which measures the linear correlation between our predicted image y and its ground truth x. Specifically, PCC is the covariance of the two variables divided by the product of their standard deviations :

$$r(x,y) = \frac{\sum_{i=0}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=0}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=0}^{n}(y_i - \bar{y})^2}},$$

where $\bar{x}$ and $\bar{y}$ are the means of the intensity datasets x and y, respectively. The higher the PCC between our predicted images and the ground truth images, the better our model performs.

Sample training loss plots are provided (3.3 Fig), reflecting the use of early stopping during training. When the validation loss did not decrease for 75 epochs, the training process was terminated. The training and test set sizes and results for all experimental conditions are provided in S1 Table, and timing data as a function of epochs is given for key datasets (see Fig 6).

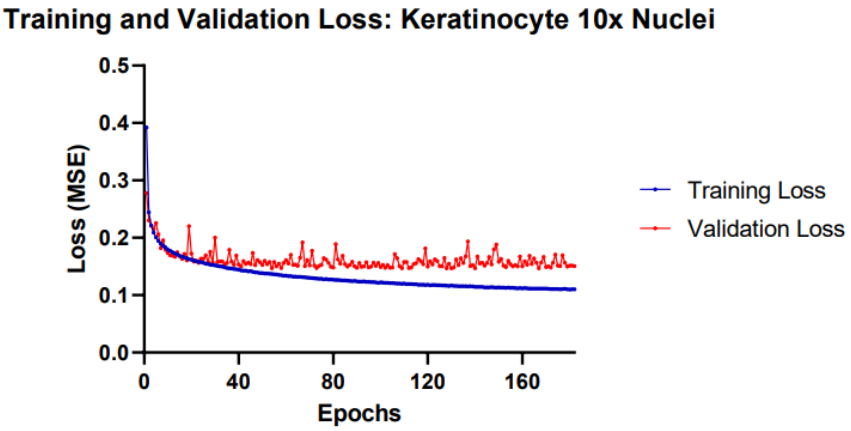

FIGURE 3.3 – U-Net architecture

3.5 Model Testing and Evaluation

3.5.1 Testing Procedure

The model testing involved several key steps :

— **Setup and Extraction** : The model was first set up by installing the necessary libraries. The architecture and weights of the model were extracted from a '.keras' file, which was initially compressed in a zip archive.
— **Model Loading and Compilation** : The architecture of the model was loaded from a 'config.json' file, and the pre-trained weights were loaded from a '.h5' file. The model was then compiled using the mean squared error (MSE) loss function and the Adadelta optimizer.
— **Evaluation and Prediction** : The model was evaluated on a test dataset to determine its performance, including test loss and accuracy. Predictions were made on the test data, and the resulting images were saved in TIFF format.

3.5.2 External Image Testing

To assess the quality of the model's predictions, the following approach was used :

— **Image Loading and Normalization** : Images were loaded from specified paths, including predicted images, ground truth images, and input images. These images were normalized for better visualization.
— **Image Visualization** : The images were converted to color for clear labeling, and labels such as "Input," "Ground Truth," and "Prediction" were added. The images were then combined horizontally and displayed to visually compare the model's predictions with the ground truth.

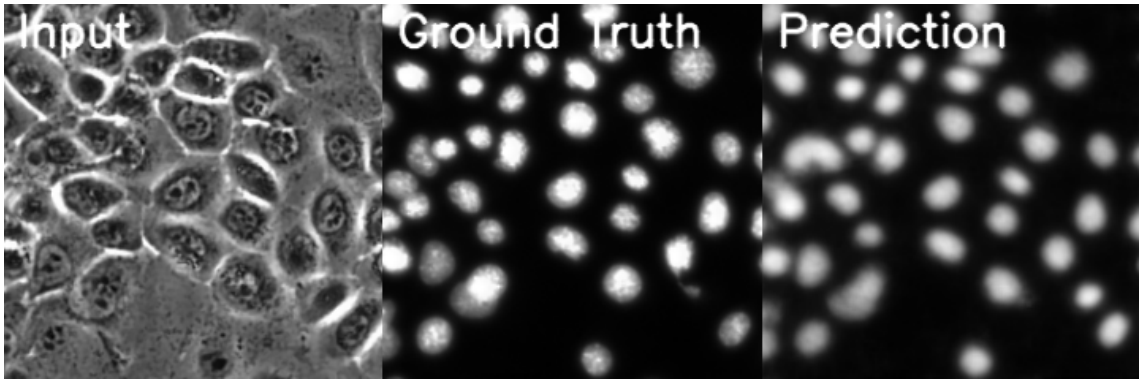

FIGURE 3.4 – DAPI generation exemple from Brightfield image input

Chapitre 4

Quantum Computing for DNA Sequence Comparison

4.1 Introduction to Quantum Computing Notation

Quantum computing harnesses the principles of quantum mechanics to perform computations that would be infeasible for classical computers. Understanding the fundamental concepts is crucial for applying quantum computing to complex problems, such as DNA sequence comparison.

4.1.1 Qubits

Qubits are the fundamental units of quantum information. Unlike classical bits, which are either 0 or 1, qubits can exist in a superposition of both states simultaneously. This ability allows quantum computers to process multiple possibilities at once, offering a significant computational advantage.

4.1.2 Superposition

Superposition enables a qubit to be in a combination of 0 and 1 states simultaneously. For instance, two qubits can exist in a superposition of four states : "00", "01", "10", and "11". Generally, n qubits can represent 2^n different states, illustrating exponential growth in complexity with the addition of each qubit.

4.1.3 Entanglement

Entanglement is a quantum phenomenon where qubits become interconnected, such that the state of one qubit directly affects the state of another, regardless of the distance separating them. This property is essential for various quantum algorithms and protocols.

4.2 Quantum Computing for DNA Sequence Comparison

Recent advancements in quantum computing have shown remarkable promise for bioinformatics, particularly in DNA sequence comparison. Our research focuses on utilizing quantum superposition to enhance the efficiency of string matching and comparison tasks.

4.2.1 Quantum Superposition for DNA Sequences

Quantum superposition allows DNA sequences to be encoded into quantum states, enabling rapid comparison and analysis. By using quantum states, we can represent DNA sequences with fewer qubits compared to classical methods. For instance, we can use just 7 qubits to encode and compare genetic codes for Yeast, Protozoan, and Bacterial, effectively reducing the computational complexity from 2^n to n qubits. This reduction significantly enhances the efficiency of the comparison process.

4.2.2 Quantum Algorithms

Our approach leverages Grover's search algorithm, which provides a quadratic speedup for string matching tasks. This algorithm is an improvement over classical search algorithms and avoids the limitations of earlier quantum algorithms that relied on memory oracles. Instead, we utilize SWAP gates, eliminating the need for random access memory and improving algorithmic efficiency.

4.2.3 Encoding and Comparing Bitstrings

To demonstrate quantum superposition, we encode genetic codes as bitstrings using 7 qubits. For example :

— YEAST : ----------MM----------
— PROTOZOAN : -MM----------M--------MMMM----------M--------
— BACTERIAL : --M----------M--------MMMM----------M--------

We use the first 6 qubits for indexing and the last qubit for content. Quantum states are represented as :

$$|YEAST\rangle = \frac{1}{8}\left(|000000\rangle|0\rangle + |000001\rangle|0\rangle + |000010\rangle|0\rangle + |000011\rangle|0\rangle + \ldots\right) \quad (4.1)$$

$$|PROTOZOAN\rangle = \frac{1}{8}\left(|000000\rangle|0\rangle + |000001\rangle|0\rangle + |000010\rangle|1\rangle + |000011\rangle|1\rangle + \ldots\right) \quad (4.2)$$

$$|BACTERIAL\rangle = \frac{1}{8}\left(|000000\rangle|0\rangle + |000001\rangle|0\rangle + |000010\rangle|0\rangle + |000011\rangle|1\rangle + \ldots\right) \quad (4.3)$$

These quantum states facilitate the comparison of genetic codes, showcasing the power of quantum superposition in bioinformatics.

4.3 Conclusion and Future Prospects

The integration of quantum computing into bioinformatics represents a significant leap forward in DNA sequence analysis. By reducing computational complexity and enhancing the efficiency of string matching tasks, quantum computing offers transformative potential. Future research will focus on refining these algorithms and exploring new applications in genomic data analysis.

Chapitre 5

Demonstration of the Prototype

Demonstration of the Realized Prototype

▶ Figure 5.1 presents an overall visualization of the virtual laboratory. It displays the different elements and instruments present in the work environment, providing an overview of the functionalities offered by the laboratory.

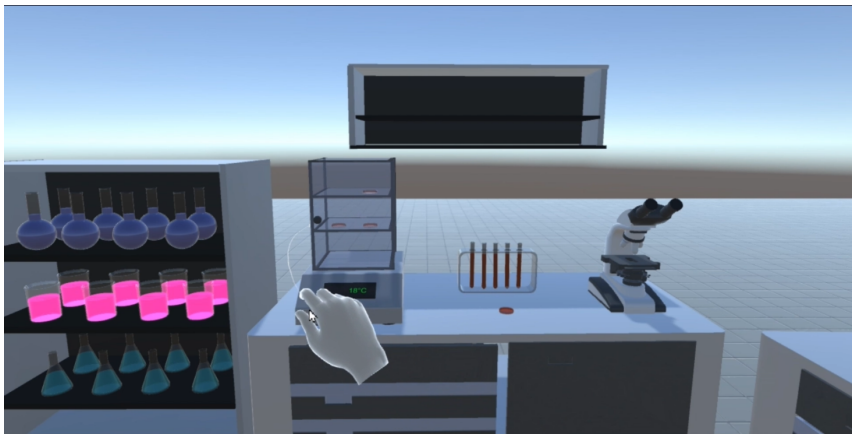

FIGURE 5.1 – General visualization of the virtual laboratory

▶ Figure 5.2 illustrates a test of the integrated alert system. The test involves detecting a refrigerator door left open, where test samples are stored. The system triggers an alert to remind the user to close the door, ensuring proper sample preservation.

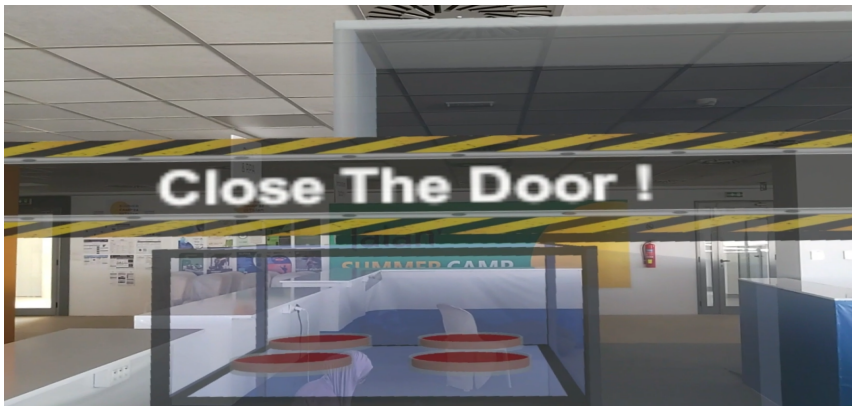

FIGURE 5.2 – Alert system test

Experiments Conducted

Adding Necessary Elements for Cell Flourishing

► After placing the sample under the electron microscope to observe the cells, the necessary elements for cell flourishing are added. This is illustrated in Figure 5.3.

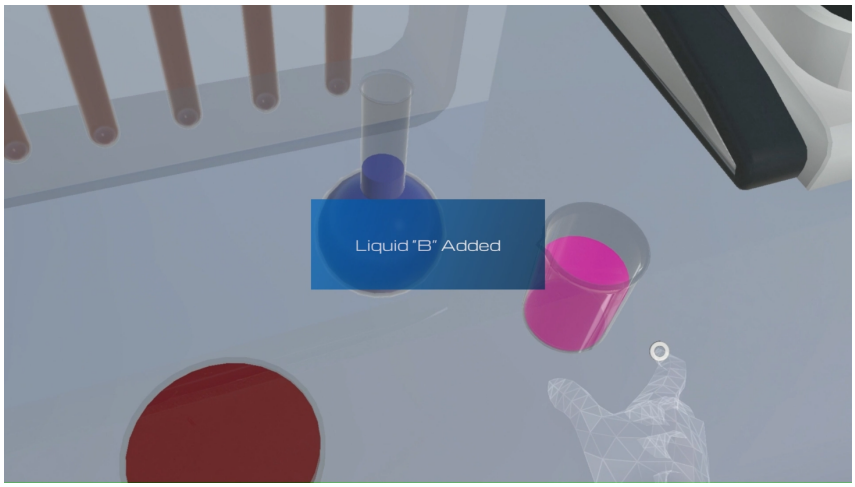

FIGURE 5.3 – Adding necessary elements for cell flourishing

► Figure 5.4 shows the final result after applying the machine learning model for cell staining.

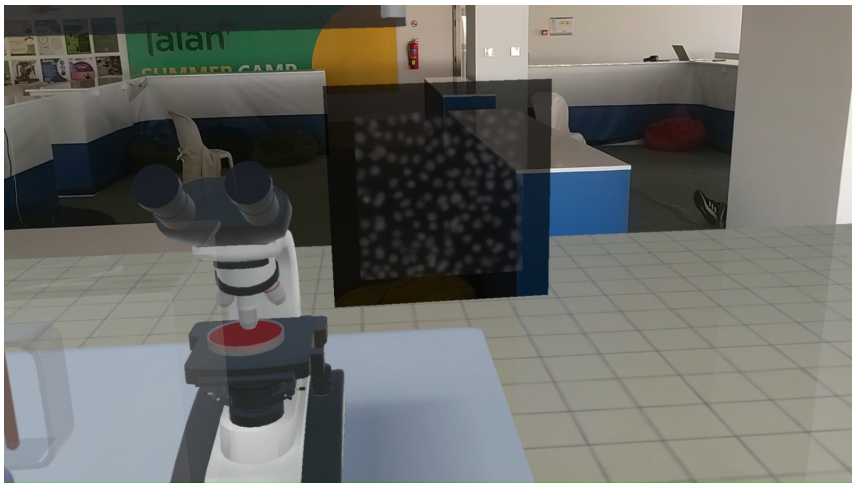

FIGURE 5.4 – Result of the first experiment : cell flourishing

s

Conclusion générale

Our LabMentor assistant promises to transform laboratories into more efficient environments, reducing both costs and time while improving the quality and accuracy of scientific results.

To enhance our solution, we plan to implement federated learning. This approach allows researchers to access research data collected from various institutions while ensuring the security and confidentiality of information.

We may also add a remote collaboration feature, enabling researchers from around the world to work together, regardless of their location.

Bibliographie