Bachelor's Degree in Bioinformatics (UPF-UPC-UB)
Final Degree Project

Application and development of a CNN model to optimize an OligoFISSEQ image obtention pipeline

Aina Martí Aranda

Scientific director: David Castillo[1], Marc A. Marti-Renom[1]

[1]CNAG-CRG, Centre for Genomic Regulation (CRG), Barcelona Institute of Science and Technology (BIST), Barcelona, Spain.

Abstract

Motivation: Imaging genomes is gaining importance due to the close relation between spatial genome organization and many biological processes including cell differentiation, DNA replication, and development. One of the latest developments in genome imaging is OligoFISSEQ, which is based on Fluorescence In Situ Sequencing technologies (FISSEQ) and uses barcoded Oligopaint probes. OligoFISSEQ is high throughput and has the potential to produce hundreds or thousands of images at high speed. In this project we have explored the application of an artificial intelligence deep learning algorithm (DLA) in the OligoFISSEQ image acquisition protocol to classify the obtained images according to their expected quality.

Results: In this project we developed and tested several DLA models to classify the images generated by OligoFISSEQ, identifying the cells that are likely to provide rich structural information. Several approaches for image preprocessing, model architecture, and data augmentation were assessed for this specific scenario. Among the various methods tested, this study achieved the best regression models with mean squared errors of 0.0149 for model 4, 0.0131 for model 3, 0.0109 for model 2, and 0.0169 for model 1, which used as binary classifiers yielded accuracies of 0.64, 0.80, 0.88, and 0.88 in the external validation dataset, respectively.

1. Introduction

Genomes do not work as linear molecules but instead, they are spatially organized and packaged inside the nucleus[2]. Chromosomes are folded and organized at different levels forming the 3D structure of the genome, which regulates many crucial biological processes such as gene expression, cell differentiation, meiosis, DNA replication, and development[2,3,4]. The 3D arrangement of the genome is very dynamic and varies depending on the cell state and type. Throughout cell differentiation, the proper organization and structures of the genome are crucial for many key molecular events. Therefore, diverse defects in these specific conformations, i.e. missorganizations, can produce developmental abnormalities, diseases, and tumorigenesis[2,3,4]. Consequently, this has carried importance to the development of tools and methods to image the 3D conformation of the genome. Images at high resolution could aid study and better characterize many human diseases.

Historically, the poor availability of tools studying the 3D genome restricted the study of chromatin architecture, leading to a huge challenge for researchers trying to study multiple diseases related to this field[2]. However, the current progress in this area and the increasing amount of 3D structure data provide an unprecedented resource that can be used to re-examine the causes of many diseases[2]. Researchers at CNAG and Harvard Medical School have developed a potentially enabling class of methods called OligoFISSEQ[1], which are the foundations of the current project. OligoFISSEQ is based on fluorescence in situ hybridization[5], used previously, for example, to demonstrate chromosome territories in interphase cells[6,7]. This new technology is a suite of three methods that use Fluorescence In Situ Sequencing (FISSEQ)[8,34] with barcoded Oligopaint[9] probes to enable the rapid 3D mapping of the targeted genomic regions inside the nucleus[1].

The usage of FISH technologies in previous studies showed the labeling of up to 40 regions to study chromosomal paths, chromatin spatial organization, A-B chromatin compartments[25,26], and the visualization of nearly entire genomes by addressing one chromosome arm at a time[1,27,28]. However, OligoFISSEQ relies on the multiplexing of the signal between different channels to maximize the number of detected targets in a certain number of sequencing rounds. Other methods that use the rounds sequentially can target up to F·N regions, being F the number of fluorophores and N the number of rounds of sequencing. Contrarily, the OligoFISSEQ strategy being discussed here has the potential to target up to F^N regions[1] while obviating the need for target amplification thanks to the incorporation of Oligopaint[9] probes. Furthermore, the usage of diffraction-limited microscopy facilitates imaging the same genomic region in thousands of cells and therefore provide the statistical power required to approach cell-to-cell variability[1]. As a result, we are in front of a likely enabling class of methods providing high throughput of images of thousands of single-cell targets, which brings us closer to whole-genome imaging explaining variability within cells. Consequently, due to the potential of the technology to generate huge amounts of data for the downstream analysis, this technology requires high levels of optimization and automatization.

Originally, the images obtained from sequencing rounds were manually analyzed, but since manual decoding is not reconcilable with high throughput, the process was automatized by creating an every-pixel automated pipeline, which was structured as a two-tier system[1]. As shown by the authors of the paper, the detection pipeline addresses the decoding of the targets by analyzing every pixel individually and grouping them to compare the different signal intensities and sizes. Tier 1, which corresponds to the preprocessing step, filters out pixels below a minimum intensity and/or patch size. Secondly, tier 2, which corresponds to chromosome tracing, progressively lowers these requirements to detect signals previously discarded in tier 1. Moreover, in this case, a distance requirement is applied: new signals from the same chromosome are required to be within 4.5um of the euclidean center of tier 1 detected targets. This new condition takes into account both the tendency of chromosomes to occupy specific territories[7] and the particular distances between targeted regions within the same chromosome. The two-tier system was shown to detect 80.2% ± 7.3% of targeted regions in each nucleus with at least 70% of targeted regions recovered in ~70% of the cells[1]. Yet, this process reveals that many cells end up with a low barcode recovery ratio and do not contribute to the final analysis.

Since this technology generates huge amounts of images representing the 3D conformational space of the genome within a cell, it requires many hours of imaging and high amounts of disk space. Presently, researchers need to closely inspect the wide-field images to select regions of interest containing cells that, in the eyes of the researcher, are likely to provide a decent amount of detected barcodes. Nevertheless, although all cells selected by the researchers are analyzed, not all of them are finally providing useful information. This means that an early automatic filtering stage could potentially save time and disk space. Additionally, another problem of this methodology is that, since samples are taken and manipulated between rounds of sequencing to carry out all the chemistry processes, the position of the microscope fields of view (FOV) need to be carefully maintained and annotated in order to return back to the same location[1]. Changes in the order of nanometers in the position of the FOV, which are very common and difficult to control, can produce the loss of valuable information as well as difficulting the posterior analysis. All this human involvement and sample manipulation adds a lot of errors and variability to the different experiments. In order to overcome all these problems, one of the current goals to improve the OligoFISSEQ methodology, as stated by the members of the lab, is to build a self-driving microscope. Such a microscope would optimize the time and disk space needed for the experiment while reducing the amount of human implication, hence lowering part of the variability between samples.

Artificial Neural Networks are gaining importance in many scientific areas and they have already made a huge impact in many fields[14,20,21,33]. In this project, as a contribution to the discussed objective, we searched for an artificial intelligence deep learning algorithm (DLA) to provide time and disk space optimization by avoiding the processing of cells that will provide scarce information (i.e. cells yielding low barcode recovery). At the same time, this algorithm automatizes the acquisition of FOV, thus reducing the level of human participation in the process. Since the images are processed and analyzed looking for shapes and peaks of intensities[1], it is logical to think that there might be some kind of pattern, easily recognizable by a convolutional neural network (CNN), that could explain the amount of information (i.e. percentage of barcode detection) that each image is providing to the experiment. Thus, allowing us to automatically drive the focus of the microscope towards regions of the plate that contain cells that are likely to provide rich structural information.

1.1 Objectives

In this project we test different image preprocessing techniques, model architectures, and data augmentation methods to identify the best procedure to classify OligoFISSEQ single-cell images using CNNs. The foremost aim is to be able to automatically discard cells at the earliest opportunity when predicted to yield low structural information.

This recently developed technology is continuously evolving and the protocols are still being improved. As a result, the models trained in this project will not be applicable to newer acquired datasets with different characteristics and properties. However, the aim of this project is not only to train the models but also to find the best procedures to deal with this type of information so that it will be straightforward to implement and train the models with newly acquired OligoFISSEQ images.

2. Methods

In this section, we detail the methods used to classify OligoFISSEQ images by implementing several Convolutional Neural Networks (CNN). This procedure used 1171 images from two datasets containing several biological and technical replicates to train, validate and test the CNN models. Features considered as key for the analysis of these images were enhanced using image preprocessing techniques. Three different approaches were tested and compared. Secondly, four model architectures of different complexities were compared to identify the one that best fitted the data size and type. Finally, in order to address the limitations of data size in this particular project, we addressed three data augmentation procedures as an approach to improve the performances of the models.

2.1 Datasets

For this project we used two datasets (36plex-5K and 36plex-1K) generated in the Wu and Marti-Renom labs. The two datasets together make up a total of 1171 cell images and consist of several biological and technical replicates (Supplementary Table 2). Since this is the first project using and developing this recent and innovative methodology, there is no more data available. Both datasets image six regions along six chromosomes (chr2, chr3, chr5, chr16, chr19, and chrX), having a unique barcode for each of those targets (Supplementary Figure 1a). All targeted regions were sequenced using ligation based identification of targets (O-LIT), which is one of the chemistry protocols that can be used for OligoFISSEQ[1]. Both datasets targeted a total of 66 regions in PGPf1 cells (six regions for each homolog of the 5 chromosomes plus six regions in the single X chromosome).

The 36plex-5K dataset targets were sequenced using the SOLiD[10] chemistry. This dataset was targeting 66 regions, each bound by a constant number of 5,000 Oligopaint nucleotide probes. Targeted regions, in this case, were ranging in size between 642kb and 1.22Mb (876kb on average). Four rounds of O-LIT were sequenced using both Oligopaint streets simultaneously. As part of the validation of the technology the same barcode was assigned to targets 3qR3 and 5pR3.

The 36plex-1K dataset was targeting the same regions and it adopted the same barcodes with the exception of 5pR3, which was given a new barcode. In this case, a new sequencing strategy named “just enough barcodes” (JEB) was designed to replace SOLiD. This strategy used only 1,000 Oligopaint nucleotide probes per region with an average of 173Kb of size by taking advantage of the universal base deoxyinosine[1,11]. Five rounds of O-LIT were sequenced assessing only one Oligopaint street. This second dataset improved the signal-to-nuclear background ratio, which also improved genomic resolution and the percentage of barcode detection.

Images from the microscope were initially aligned, segmented, and deconvolved using 20 iterations of the Richardson-Lucy algorithm to improve their resolution[40] (Supplementary Figure 2). After those modifications, the images were analyzed by the every-pixel pipeline to identify and decode the barcode information. For the development of this project, we will use the already deconvolved, aligned and segmented images (a total of 1171 cells).

2.2 Data splitting

From the whole dataset with a total of 1171 cell images, data splitting was performed to separate the training, internal validation and external validation datasets. The complete biological replicate OFQv69, which consisted of 125 representative cell images from all cases, was separated to be used as the external validation (Supplementary Table 2). The remaining 1046 cell images from the other replicates were used as the training dataset. Those images were randomly splitted into 4 different folds to perform cross validation (CV)[17].

Each iteration of CV used one fold of 261 or 262 images as the validation set while the other folds were used for the training process. This resulted in training sets of 785 or 784 images depending on the folds used as validation sets (Supplementary Figure 3). The images used in each fold were the same for all models developed in this project.

Then, the performances were estimated by averaging the performances over folds. In this project, the metric used to quantify the performances of the models was the mean squared error, as stated in section 2.12.

2.3 Data structure

The images used in this project were the output of 4 (36plex-5K) and 5 (36plex-1K) rounds of sequencing where the targets in each cell were visualized in the 3D space. At each round, the cells were imaged at five different wavelengths of colors, representing the four color coding for the barcodes plus the staining of the nucleus: Alexa Fluor 647, Texas Red, Cy3, Alexa Fluor 488 and DAPI (Supplementary Figure 1b). The cells were also imaged along the axial dimension in a series of z slices. Hence, the images used in this project consisted of 5 different dimensions representing the x and y axis, the channels imaged, z-slices, and finally the rounds of sequencing.

2.4 Data visualization

The images used in this project were previously processed using ImageJ/FIJI[29] to align and segment the cells[1]. Moreover, in this project we also used ImageJ/FIJI, and matplotlib[30] (v3.3.4) to analyze and visualize the cells individually.

2.5 Data labels

In this project, we aimed to classify the images in accordance with the percentages of barcodes detected with each of them. The labels ranged between 0 and 1, being 1 the 100% of barcodes identified in a specific cell.

The labels were obtained by applying the every-pixel pipeline[1] to the images given to the models. The number of barcodes detected with this pipeline was used to compute the percentage of barcode recovery.

2.6 Models built

As stated previously, the foremost aim of this project was to discard cells at the earliest opportunity. Given that the rounds of sequencing are obtained progressively in the OligoFISSEQ methodology, we repeated all our tests to train four different DLA models. The first model tried to discard the low recovery cells with only the images from the first round of sequencing. Once the second round was already sequenced, the second DLA used the new information together with the previous one to make a better prediction and discard the cells that were likely to yield a low recovery ratio at this new step. Finally, we did the same with the next two rounds of sequencing and their corresponding DLA models.

For the sake of simplicity, we will refer to those models as model 1, model 2, model 3, and model 4 corresponding to the number of rounds of sequencing that they are taking into account.

2.7 Convolutional Neural Networks

The methods being deployed in this project focus on Convolutional Neural networks (CNN), a type of DLA used to work with images searching for complex and hierarchically organized patterns[15]. This type of neural network falls into the discriminative category, which means that it follows a bottom-up approach in which data flows from the input layer via the hidden layers to the output layer, using supervised training[15].

This type of network usually consists of three types of layers: convolutional and pooling, which perform feature extraction, and fully connected layers, which generate the final output[14,35]. Usually, convolutional and pooling layers are alternated in the first part of the model and are followed by one or several fully connected layers at the end. Lastly, those models require a final layer where the number of neurons matches the number of outputs that need to be obtained (i.e. the probability for each class or one single neuron when building a regression model[14]). The order of these layers and their corresponding parameters define the model architecture. The more layers, the more complex patterns the CNN will be able to identify. However the lowered number of neurons and layers usually reduces the training time and also the probability of overfitting[15,18].

2.8 Model architecture

In this project, several model architectures of different complexities were tested and compared in order to find the best model architecture that fitted the data used. All of them consisted of several pooling and convolutional layers alternated between them. Their main difference focused on the number of layers and the number of neurons in each layer, which defined their complexity. Among the different architectures addressed, the one leading to the best model performances (Figure 1) consisted of: one convolutional layer with 32 neurons, two convolutional layers of 64 neurons, two convolutional layers of 126 neurons and two fully connected layers of 32 neurons with a pooling layer in between each of those groups of layers. Finally the architecture had a final dense layer of 1 neuron to generate the final output of the regression model (i.e. the percentage of barcode recovery per image).

In all convolutional layers and the two fully connected layers of 32 neurons, the activation function applied was the Rectified Linear Unit (ReLU) as shown in equation 1. Activation functions are used because of their easy computable partial derivatives of the error[15]. This particular function applies a nonlinear transformation to the given values, and is currently the most commonly one[14]. On the contrary, we used the sigmoid activation function for the final layer. This nonlinear transformation (equation 2) narrows the range of outputs to 0/1. This activation function fulfilled the model requirements since the aim of the model was to predict the fraction of barcodes that would be identified with each given image.

$$f(x) = max(0,\ x) \qquad (1)$$

$$f(x) = \frac{1}{1+e^{-x}} \qquad (2)$$

2.9 Image preprocessing

In the implementation of CNNs it is crucial to preprocess the images in order to prepare them for the model. In this project, we applied image preprocessing to normalize the image dimensions across samples and do some modifications to enhance the key features. For the sake of feature enhancement and noise reduction, we applied pixel modifications such as thresholding[36,37] and normalization. The image preprocessing step is crucial for any CNN model development since it has a strong impact in the model performance[37,38]. So as to find the best image preprocessing procedure, it is important to understand the images and accurately identify the features that need to be enhanced. We designed different image preprocessing methods that could fit our image requirements and we compared their performances. Among the several approaches addressed, the one reaching the best results consisted of 6 different steps.

In the first step of the procedure, all images in the z-slices were projected by selecting the maximum intensities at each pixel position. This was done in order to reduce the complexity of the data that was

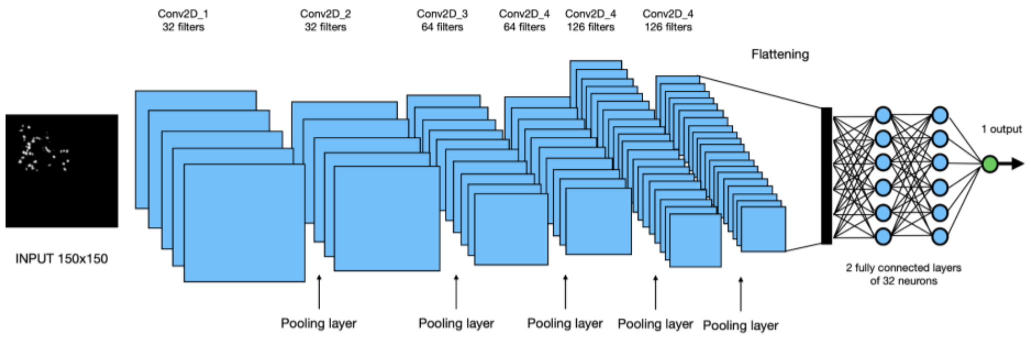

***Figure 1**. Best performing model architecture used in this project.*

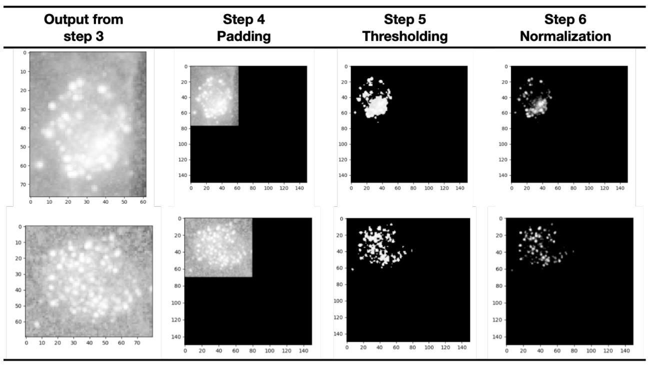

Figure 2: *Image preprocessing part 2. Images from the first round of sequencing (channels and z-slices projected with maximum intensity) at each step of the image preprocessing procedure used. Images from the first column correspond to the output from step 3 of the method. Next step applies padding to normalize the size and shape of all images to 150x150 pixels. Step 5 applies thresholding, and finally step 6 applies a 0/1 normalization of the pixels different to 0.*

given to the model while keeping the most important information. At the same time, since models need to receive all the images with the same dimensions[13,31], we used this step to overcome the variability produced by the non-constant number of z-slices. In addition, the channel showing the DAPI staining of the whole nucleus was discarded together with the fifth round of sequencing from the second dataset.

In the second step of the procedure, we normalized all the pixel values across channels, converting their values to a range between 0 and 1. Subsequently, in the third step, we projected the channel images using again the maximum intensities, which resulted in all samples having one image per round of sequencing.

After this, zero padding was applied to the images in the fourth step. This process refers to the addition of pixels with value 0 to the borders of the image[31]. In our case, padding was added to the right and bottom sides, leaving the original image at the left-top corner. This step was applied with the only aim of resizing and reshaping the images. Given that the conventional cropping approach was not applicable in this case (due to the loss of information that it would imply), we chose to apply zero-padding. Moreover, zero padding does not affect the performance of the models while it suppresses the risk of deforming the key patterns[31].

Afterwards, the fifth step applied the thresholding technique[36]. In the images used, we were interested in the patches of highest intensity of the images, which had a higher probability of belonging to a barcode. In order to enhance those pixels, the thresholding step set to 0 all the pixel intensities that fell below the specified threshold. In this case, we applied a threshold of 0.90 to the images from the 36plex-5K dataset, and a threshold of 0.87 to the samples from the 36plex-1K dataset. Such difference was due to the fact that the second dataset used less oligopaint probes (1K vs 5K) and addressed target regions of shorter length, thus resulting in patches with lower intensity in general. The values used were based on the same values applied in the every-pixel pipeline when it filters the patches by intensity. However, in this case, we lowered the thresholds in order to reduce the strictness of the method.

Finally, the last step of the image processing procedure applied a normalization to the pixels that were left with high intensities in order to enhance the differences between them. (Figure 2)

To apply this process we used numpy[12] (v1.19.2) and pandas (v1.2.1), which are open-source data analysis and manipulation tools for the Python programming language. We used numpy to store the images (lists of pixel values) and the image labels, and pandas to store the different information associated with each cell.

2.10 Data augmentation

The training of any Artificial Neural Network requires lots of data samples in the training dataset in order to achieve a good performance[32]. In this project we addressed several data augmentation procedures as an approach to overcome the problem of data scarcity. This process creates new artificial images based on the original data by performing several random modifications such as rotation, flipping, thresholding and others[23]. Every copied image, even though it is based on the original one with simple modifications, has a unique pixel value distribution, which allows the model to learn from new examples and learn meaningful patterns from them[19]. Models trained with few samples are prompt to detect the noise and specific characteristics of the example images instead of focusing on the real meaningful patterns. Therefore, the generation of new artificial examples usually helps the model to achieve best performances on new datasets, thus it improves its robustness. Among the different modifications that could be applied to the images, we played with the variations that were not affecting nor altering the main key information of the image. Hence, we generated

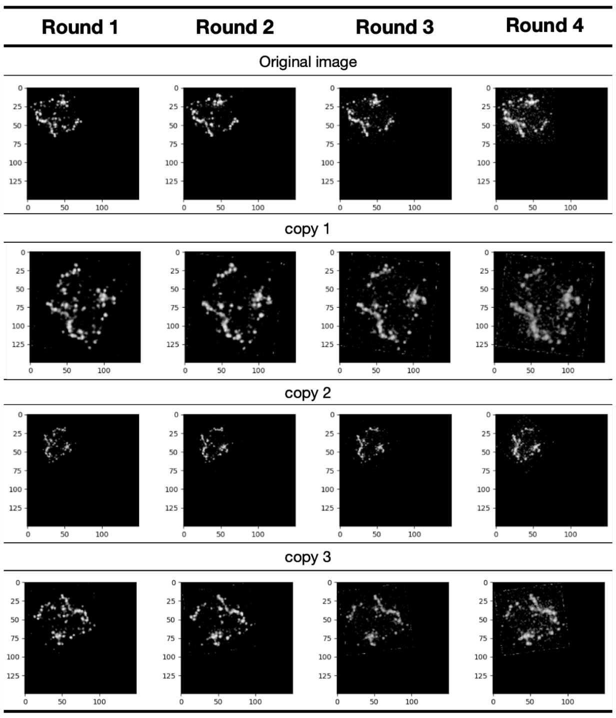

Figure 3: *Data augmentation. Three examples of augmented images. First row shows the original image after passed through the image preprocessing procedure 1. Next rows show the same cell images with several modifications. Rotations, size modifications and reflections were applied.*

new artificial images without introducing unrealistic changes that would not be seen in new real images.

As a result, we built three different approaches using different types of modifications. Finally, the one leading to the best performances carried out 3 simple modifications: flipping/reflection, rotation, and size modification. For each original image, we generated 9 copies with different alterations, thus ending with 10 images from the same cell. For each copy, random values were generated to determine the angle of rotation, and the level of size modification while never generating bigger images than 150x150 pixels (Figure 3). Finally, only 60% of the copies were flipped.

This procedure was applied only to the training set in order to generate new images. The validation data was not augmented nor modified given that it was used to test how the model would perform in real world data.

2.11 Model training

The parameters of the model that can be learned, which are the so called weight values, are trained by using some mathematical methods such as backpropagation and gradient descent[20,35], which efficiently explore the space of solutions and return the optimal set of weight values that will lead to the best possible solutions[24]. This optimal set of parameters is found when the cost function reaches its minimum value[15]. This training process is performed by forcing the model to learn from various image examples that are already labeled. Therefore, the more examples in the training set, the more robust the model will be towards new acquired data, thus reducing overfitting[32].

To build and train the models we used the python open source library Tensorflow13 (v2.2.0) and Keras, the most used library for Deep Learning development in python. To train the models, we used the optimizer Adam[16,22], which is a general-purpose system since it uses a stochastic gradient descent method that can train the model via back-propagation. According to Kingma et al., 2015, the Adam optimizer method is "computationally efficient, has little memory requirement, is invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/ parameters"[16,22].

The loss function used was the mean squared error as shown in equation 3, which computes the mean of squares of errors between labels and predictions.

$$MSE = \frac{1}{n} \cdot \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \qquad (3)$$

In order to prevent overfitting, we applied the early stop callback tool offered by keras. This function takes as argument the number of epochs after which the process will stop training if the monitored metric has stopped improving, as it can be understood from the documentation of the package[13].

2.12 Model performances

The performances of the models were quantified by using the loss function mean squared error as shown in equation 3. This metric applied to the validation datasets has been used to compare the model performances between the different approaches performed. Each model tested was repeated for all folds in order to apply cross validation, and the final performance comparisons were made with the mean values across the folds.

In this project, the usage of the area under the curve (AUC) for model comparison as binary classifiers was not recommended due to the small dataset size. Small sample sizes produce stepped ROC curves, which make each sample have a high contribution to the sensitivity and specificity measures[39], thus seeing a huge variability in the AUC value with small changes in the predictions. However, the ROC curve was used to compute the optimal class separator threshold for the final binary classifiers.

After identifying the best models (i.e. the ones with lowest average loss value across folds), the model built from the fold with best performance was taken to carry out the external validation test. In this test, we computed the mean squared error and the accuracy for the classification. We binarized the labels used for each model by applying a threshold of 0.5 for model 4, 0.6 for model 3, 0.7 for model 2, and 0.25 for model 1. On the contrary, the thresholds used with the prediction scores corresponded to the optimal thresholds defined by the ROC[39] curve as mentioned previously. Binarizing both sets of values we computed the accuracy of the models and used this metric as the final quantification of the performance of the models to make a binary classification.

2.13 Code script

All code generated and used in this project is available at the following link: https://github.com/ainamarti/FDP_bioinformatics.

3. Results

Model	Average MSE	Optimal threshold	accuracy	TPR	TNR	Approach
1	0.00993	0.209	0.766	0.759	0.770	A15
2	0.01093	0.762	0.797	0.795	0.806	A9
3	0.01375	0.708	0.858	0.854	0.892	A9
4	0.01475	0.614	0.821	0.819	0.825	A9

Table 1: *Performance values from the best working models when tested against the internal validation sets.*

In this project, applying the proposed model architectures, image preprocessing methods and data augmentation procedures, 15 different model approaches were evaluated for each of the four models assessed. The results for the best performing models have been presented in Table 1. Moreover, the results from all approaches tested have can be found at Supplementary table 1. The best performing approach found used image preprocessing procedure 1, model architecture 3, and data augmentation method 1, which have been explained in detail in the methods section. The combination of these methodologies achieved an averaged mean squared error of 0.014749 for model 4, 0.013753 for model 3, 0.010929 for model 2 in the CV internal validations. In contrast, most of the approaches tested for model 1 did not perform as expected retrieving the same output for all samples in the internal validation and training tests, as discussed in the next section. Between the working approaches, the best performance showed an MSE of 0.00993 by using image preprocessing procedure 1, model architecture 1, and data augmentation 1. Plots showing the performances mentioned are shown in Figure 4.

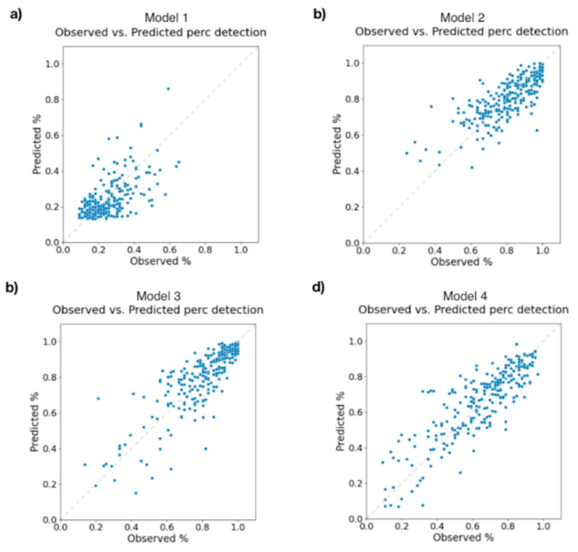

Figure 4: *Performances of the final selected models. Each one show the observed percentages of barcode detected vs the predictions made by the regression models. Each individual dot in the presented plots represent a single cell.*

For validation purposes, these models were tested against the external dataset and the results have been presented in Table 2. In addition, the visual representations for the external dataset are also shown in Figure 5.

Finally, by binarizing the labels and predictions, we tested the performance of these models as binary classifiers. For this we computed accuracies, true positive rates (TPR) and true negative rates (TNR). Results are shown in tables 1 and 2 together with the previously mentioned metrics. The final models selected achieved accuracies of 0.766, 0.797, 0.858, and 0.821 in the internal validation dataset and accuracies of 0.640, 0.800, 0.880, and 0.880 in the external dataset for models 4, 3, 2, and 1 respectively.

Model	MSE	accuracy	TPR	TNR	Approach
1	0.01690	0.640	0.638	0.642	A15
2	0.01090	0.800	0.785	0.851	A9
3	0.01310	0.880	0.886	0.800	A9
4	0.01490	0.880	0.879	0.888	A9

***Table 2**: this table shows the performance values from the best working models when tested against the external validation sets.*

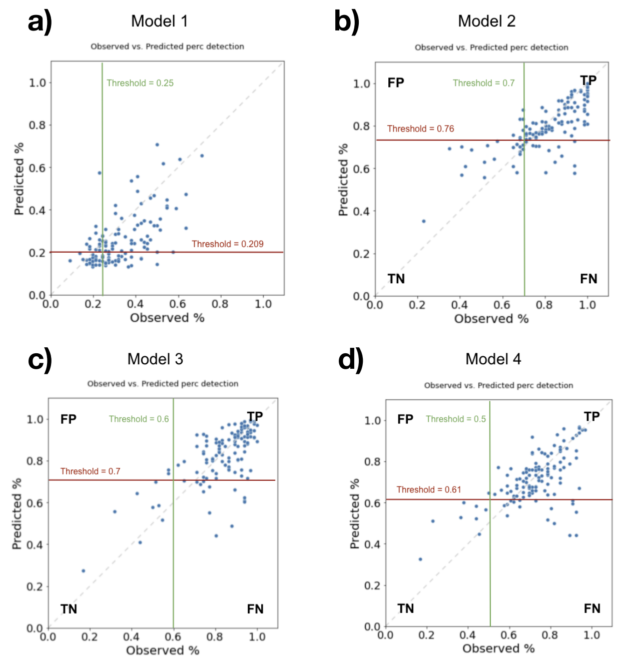

***Figure 5:** Performances of the final selected models when tested against the external dataset. Each one show the observed percentages of barcode detected vs the predictions made by the regression models. Each individual dot in the presented plots represent a single cell. Horizontal lines presented in red show the optimal threshold computed from the AUC with the internal validation*

Lastly, responding to the foremost aim of this project, we applied the four models to test the progressive exclusion of cells (Figure 6). Moreover, due to the low precision of model 1, we tested the progressive riddance of cells without the usage of this first model, and finally compared this to the usage of model 4 alone.

4. Discussion

The models built and the performances achieved demonstrate that, despite data scarcity, it is possible to build regression models to explain the amount of structural information contained in each image. Yet, with the availability of more input data, the models could be potentially improved in the future. Here, we discuss the results collected in this project and potential methods to refine the methodology in further explorations.

So as to compare the achievements of the models, we compared the Mean Squared Error (MSE) values between the different approaches, which computes the differences between predicted and observed values, as detailed in the methods section 2.11. Since the labels used for each of the four models built had different distributions of values, the final MSE values should not be compared between models 1, 2, 3 and 4.

In the results obtained, we could detect that some model approaches failed to work as expected (Supplementary table 1). Those models produced invalid results, giving the same values for all samples in the training and the internal validation sets. The reasons for the failure of these models to predict accurate values and detect key patterns in the input data could be related to the usage of wrong techniques for the model, or the poor relation between the inputs and the labels. When the labels used for the DLA are not fully explanatory of the information contained in the images, it is extremely complex to identify the patterns that define the relation between them.

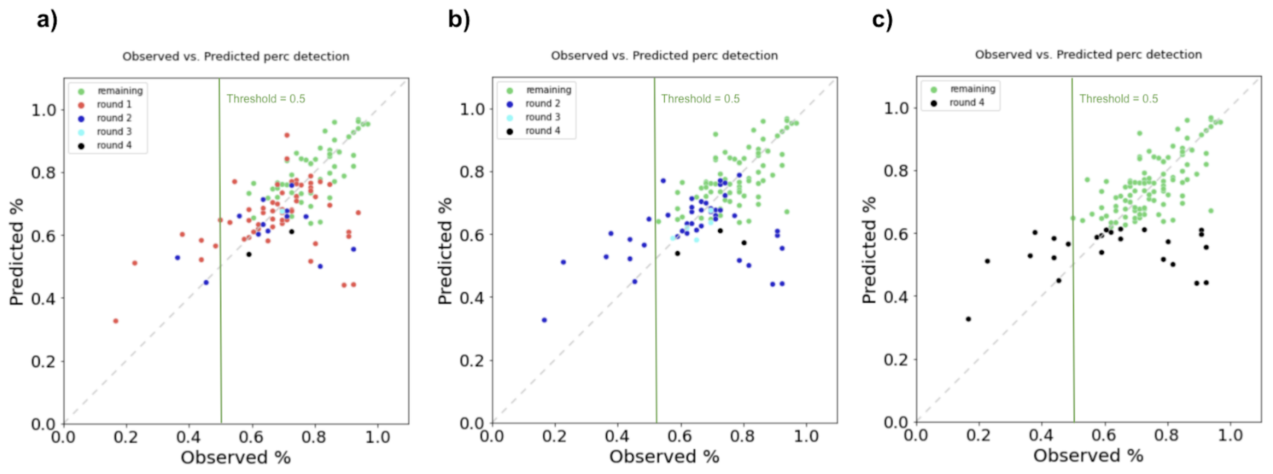

***Figure 6:** Cells discarded at each round of sequencing.*

Therefore, in those cases, the model is prone to find a suboptimal solution where all samples get the same output, being this one close to the average of the labels. In addition, weight initialization could also play a key role in this phenomenon, which can drive the model towards this suboptimal and very simple solution.

By building two models with the same parameters (approaches 1 and 2) but changing the final size of the images, we tested the effect of image padding for the model performances. Here, we confirmed that it was not affecting the final model achievements since we did not observe a significant change in MSE values. This stems from the fact that neighboring zero pixels do not add new information for the model, consequently they do not contribute to the identification of the key patterns while not hindering the process. Since the percentage of padding added depends on the size of the cell, we also tested the correlation between the errors of the models and the areas of the cells. We obtained values of -0.090 ($P<0.147$), -0.0004 ($P<0.995$), 0.105 ($P<0.09$), and 0.098 ($P<0.12$) in the Pearson's correlation coefficient test for the correlation between those two variables (Supplementary Figure 4). This confirms that the accuracy of the model does not depend on the amount of padding added to the image. With this background, we decided to continue our project resizing all images to 150x150 since it allowed us to accommodate all cell shapes and sizes present in the datasets while reducing execution time.

Thereafter, the next models showed how we could improve the performances by increasing the complexity of the model architecture. Third approach, which used the least complex architecture was the worst performing, being followed by approach 1, 3 and 4, which progressively increased architecture complexity and improved the performance, ending with approach 4 retrieving the best MSE results of 0.01420, 0.01505, and 0.01599 for models 2, 3, and 4. Contrarily, model 1 only achieved valid results with architectures 1 and 2, being 1 the best performing one with an MSE of 0.01037. Finally, approach 5, which used the most complex architecture, showed worse performances as compared to the previous approach, with the exception being model 2. This reveals the potential limit of model architecture complexity to allow sophisticated pattern recognition while reducing the effect of overfitting. However, since our problem of data scarcity could be masking the need of bigger architectures for better performances, we can not guarantee that the best model architecture found here is the best one for the solution of this problem. Nevertheless, we can use this as a base for further experiments.

Subsequently, once identified the best model architecture, we tested several image preprocessing techniques. Among the approaches tested, we saw the large impact that this step had on the performance of the final models. Firstly, we tested the approach detailed in the methods section 2.9 (approach 4), which at the end resulted in the best performances seen for the models as discussed previously. For the purpose of finding better procedures, we examined image preprocessing methods that applied distinct simplifications of the data. Higher simplification of the images, which projected the round images, was tested (image preprocessing 2). This test yielded invalid results since the models forecasted the same output values for all samples in every model and fold assessed. Thus, we could straightaway discard this procedure while understanding that the key information for the classification of those images was centered on the specific information at each round.

With this background in mind, we tested the opposite. In this case (approach 7), we removed the z-slices projection to keep more information for the models (image preprocessing 3). Contrarily to the previous system, this time the models could find a way to properly classify the images with the exception of model 1. Yet, this approach resulted in worse performance than earlier models. Considering the fact of using more complex images as input, we tested the same approach with a higher complexity architecture (approach 8). However, we did not see an improvement as compared to the prior method. Nevertheless, since more complex images need more training data, we propose to repeat this approach when having more data available to examine if the cause of poor performance was due to data scarcity.

In that behalf, we proceeded the experiments using model architecture 1 and image preprocessing procedure 1. Due to our limitation in the number of images available for the training process, we addressed several data augmentation procedures as an approach to improve the models. We started by generating 10 and 20 copies of each image with very simple modifications (approaches 9 and 10):

rotation, flipping, and resizing. The results of the approach using 10 copies improved the performances seen previously, showing MSE averaged values of 0.01093, 0.01375, and 0.01475 for models 2, 3 and 4. Therefore, in this case, the generation of new artificial examples helped the model to achieve better performances. On the contrary, the second approach generating 20 copies did perform very similarly but not better.

So as to search for better models, we tested other data augmentation procedures with more modifications (approaches 11, 12, 13 and 14) such as blurring, denoising, and sharpening. However, those image variations did not help the models, revealing that they could have been altering the key patterns of the images, leading to worse results. We could see that the random change of the threshold used at step 5 was highly downgrading the model performances. This effect could be explained by the fact that the constant threshold used for the images removes the background noise at the same level for all the images in the dataset. By randomly modifying the threshold of intensity, we are changing the amount of noise present in every sample. Since the noise acts as a key pattern for the classification of the images, such modification of the threshold could have prevented the correct recognition of common patterns resulting in worse performances. The results from approaches 13 and 14, which applied the same alterations without modifying the threshold, confirmed this hypothesis by revealing better performances.

Finally, since model 1 didn't return valid results for none of the models applying data augmentation, we repeated approach 1 including data augmentation procedure 1 (approach 15). As a result, we obtained the best performances seen for model 1 with MSE value of 0.00993. In the light of these results, we can conclude that data augmentation procedure 1, which has been explained in detail section 2.10, was the best performing one.

Lastly, as already mentioned, the best approach for each model was selected and tested against the external dataset. As it can be seen from the results of tables 1 and 2, model 1 was the worst performing one with a big difference as compared to the other models. This model also showed low capacity to generalize towards a new external dataset. Seeing the differences in MSE, accuracy, TPR and FPR between internal and external validation sets for model 1, we can confirm an overfitting effect produced by the model. Moreover, by making a visual inspection of the plots resulting from model 1 (Figures 4a and 5a) we can see that the performance of this model showed higher levels of error as compared to the other models when taking into account the distribution of observed percentages. This observation can be confirmed with the low accuracies presented in tables 1 and 2 for this model. Contrary to this, models provided with more information such as models 3 and 4 gave better results.

Bearing in mind these results, a possible failure of the labels used for the first models was analyzed. As it is known, the signal needed to detect the final barcodes is being added progressively throughout the rounds of sequencing. This means that it may not be easy to predict the percentage of barcode detection only with the first rounds of sequencing since we are missing most of the information. By only analyzing the images from the first rounds, several patches of the same channel might appear close together, driving the method to detect it as a single barcode patch. This drives the pipeline to retrieve low barcode recoveries, as it can be seen in figure 4a, while indeed more barcode patch digits are identifiable. Only by the addition of more round images, we can improve and tune the barcode identification process, which will reach the optimal state when using all the rounds of sequencing designed for the experiment. Moreover, with partial information, many barcodes remain not unique. This presents a challenge for the usage and performance of the first models constructed in the project. Here, as a potential future improvement, we suggest the usage of the number of patches detected in each image, taking into consideration the average patch size in the dataset, being independent of the number of targeted regions or uniqueness of the barcodes.

Once the models were constructed, we proceeded to make a visual examination of the images and their predictions. As mentioned earlier, the results obtained through the different approaches hinted that the decisive information was somehow hidden into the images obtained at each round. By analyzing the results and images from model 4, we could detect that numerous cells yielding low barcode recovery and low percentage predictions showed high noise levels and low signals at round 4. As we can understand from the methodology used in OligoFISSEQ, progressive decrease in the signal and increase in the noise is already expected. This occurrence is, in part, a consequence of the low efficiency of ligation methods together with the lack of efficiency in the fluorophore removal process. This effect is expected to happen progressively through rounds. However, it has been detected to happen abruptly in the last round of multiple cells (Supplementary Figure 5). Moreover, there have also been detected a punctual increase of background noise at round 2, and a decrease of signal in multiple cells at this same round of sequencing (Supplementary Figure 6). With the methodology used in this project and the analysis made, we constructed a way to identify this type of behavior in sequenced cells. Since this effect might be potentially caused by errors in the chemistry protocol or wrong sample manipulation, recognizing this type of defects in the images would allow researchers to promptly identify errors in the process and act accordingly.

These findings and the previous results reveal that the low quality in some round images can be a potential determining factor for low barcode recognition, while demonstrating that it may not be possible to detect defective cell images in the early steps of the method due to the lack of information.

Regarding the progressive exclusion of cells, as shown in figure 6a, it can be seen that most of the wrong cell riddance happens in round 1. As discussed earlier, this confirms the bad precision of this model. In addition, the same applies for models 2 and 3, yet those reveal better results.

To reduce the downgrading effect of model 1, we tested the progressive riddance of cells without the usage of this model (Figure 6b). In this case we can see fewer false negatives, while also detecting that all true negatives were already discarded at round 2. Thereafter, when comparing to the usage of model 4 alone, we can see a decrease in the number of false negatives.

All things considered, both methods offer different advantages. Despite the better accuracy of the usage of the last model alone, the usage of models 2, 3, and 4 together would allow the exclusion of deficient cells in the early stages of the process, thus providing higher time optimization. Nevertheless, it would also mean the loss of potentially useful information. Given this evidence, lab researchers could apply each method depending on their specific needs in each experiment.

Furthermore, taking into account the data limitation and the performance improvement achieved with data augmentation, the project shows the potential of improving these models with more data and less limitations. Ultimately, in case the first models do not achieve better performances in future experiments, model 4 can be potentially used alone for time and disk optimizations at the end of the sequencing experiments. With only this model, researchers could immediately detect errors in the process without the need to wait for results that take hours of computational analysis. Moreover, it would provide a way to keep track of the number of sequenced cells with rich structural information. Finally, it could optimize the disk space used by automatically discarding the cells predicted to yield low structural information, which at the same time would avoid unfruitful analysis time.

5. Conclusion

In this project, as a contribution to the construction of an OligoFISSEQ self-driving microscope, we searched for an artificial intelligence deep learning algorithm (DLA) to provide time and disk space optimization while reducing the amount of human intervention in the acquisition of FOV. The results obtained demonstrate that, despite data scarcity, it is possible to build regression DLA models to explain the amount of structural information contained in the cell images generated with the OligoFISSEQ technology. Moreover, they demonstrate the potential of the methodology to be improved in future investigations with less data limitations.

Despite the poor performance of the first model, the work developed in this project enables the automatic riddance of cells even at the second round of sequencing. At the same time, by the usage of the last model alone at the end of the experiments, researchers could also achieve disk space and analysis time optimization.

Yet, since this technology is continuously evolving and the protocols are still being improved, the models provide the first starting point for future investigations for the search of improved methodologies for OligoFISSEQ cell image classification.

6. Supplementary material

The supplementary material generated in this project is available at the following link: https://drive.google.com/file/d/1jgr3RRLUDLpgk5CJOgQpC3q-KmvbzaPu/view?usp=sharing

7. Acknowledgements

I would like to express my thanks to David Castillo and Marc A. Marti-Renom for their patience and guidance throughout all the steps of this project. Also, I wanted to express my gratitude to the researchers at Ting Wu lab from Harvard Medical School for the data supply and their help and guidance towards understanding OligoFISSEQ technologies.

In addition, I would like to extend my thanks to all the members of both labs for giving me the huge opportunity to work and learn with all of them.

8. References

1. Nguyen, H. Q., Chattoraj, S., Castillo, D., Nguyen, S. C., Nir, G., Lioutas, A., … Wu, C. ting. (2020). 3D mapping and accelerated super-resolution imaging of the human genome using in situ sequencing. Nature Methods, 17(8). https://doi.org/10.1038/s41592-020-0890-0
2. Zheng, H., & Xie, W. (2019). The role of 3D genome organization in development and cell differentiation. Nature Reviews Molecular Cell Biology. https://doi.org/10.1038/s41580-019-0132-4
3. Krumm A, Duan Z. Understanding the 3D genome: Emerging impacts on human disease. Semin Cell Dev Biol. 2019;90:62-77. https://doi:10.1016/j.semcdb.2018.07.004
4. Pombo, A., & Dillon, N. (2015). Three-dimensional genome architecture: Players and mechanisms. *Nature Reviews Molecular Cell Biology*. https://doi.org/10.1038/nrm3965
5. Hu, Q., Maurais, E. G., & Ly, P. (2020). Cellular and genomic approaches for exploring structural chromosomal rearrangements. *Chromosome Research*. https://doi.org/10.1007/s10577-020-09626-1
6. Bolzer, A., Kreth, G., Solovei, I., Koehler, D., Saracoglu, K., Fauth, C., … Cremer, T. (2005). Three-dimensional maps of all chromosomes in human male fibroblast nuclei and prometaphase rosettes. PLoS Biology, 3(5). https://doi.org/10.1371/journal.pbio.0030157
7. Cremer, T., & Cremer, M. (2010). Chromosome territories. Cold Spring Harbor Perspectives in Biology. https://doi.org/10.1101/cshperspect.a003889
8. Ke, R., Mignardi, M., Pacureanu, A., Svedlund, J., Botling, J., Wählby, C., & Nilsson, M. (2013). In situ sequencing for RNA analysis in preserved tissue and cells. *Nature Methods*, *10*(9). https://doi.org/10.1038/nmeth.2563
9. *Beliveau, B. J., Joyce, E. F., Apostolopoulos, N., Yilmaz, F., Fonseka, C. Y., McCole, R. B., … Wu, C. T. (2012). Versatile design and synthesis platform for visualizing genomes with Oligopaint FISH probes. Proceedings of the National Academy of Sciences of the United States of America, 109(52).* https://doi.org/10.1073/pnas.1213818110
10. Metzker, M. L. (2010). Sequencing technologies the next generation. *Nature Reviews Genetics*. https://doi.org/10.1038/nrg2626
11. Watkins, N. E., & SantaLucia, J. (2005). Nearest-neighbor thermodynamics of deoxyinosine pairs in DNA duplexes. Nucleic Acids Research, 33(19). https://doi.org/10.1093/nar/gki918
12. Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*. https://doi.org/10.1038/s41586-020-2649-2
13. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., … Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016*.
14. Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. Insights into Imaging. https://doi.org/10.1007/s13244-018-0639-9
15. Shrestha, A., & Mahmood, A. (2019). Review of deep learning algorithms and architectures. *IEEE Access*. https://doi.org/10.1109/ACCESS.2019.2912200
16. Chilimbi, T., Suzue, Y., Apacible, J., & Kalyanaraman, K. (2014). Project ADAM: Building an efficient and scalable deep learning training system. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2014*.
17. Xu, Y., & Goodacre, R. (2018). On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning. *Journal of Analysis and Testing*, *2*(3). https://doi.org/10.1007/s41664-018-0068-2
18. Ying, X. (2019). An Overview of Overfitting and its Solutions. In *Journal of Physics: Conference Series* (Vol. 1168). https://doi.org/10.1088/1742-6596/1168/2/022022
19. -Shanthi, P. B., Faruqi, F., Hareesha, K. S., & Kudva, R. (2019). Deep Convolution Neural Network for malignancy detection and classification in microscopic uterine cervix cell images. *Asian Pacific Journal of Cancer Prevention*, *20*(11). https://doi.org/10.31557/APJCP.2019.20.11.3447
20. Lecun, Y., Bengio, Y., & Hinton, G. (2015, May 27). Deep learning. Nature. Nature Publishing Group. https://doi.org/10.1038/nature14539.
21. Sabanayagam, C., Xu, D., Ting, D. S. W., Nusinovici, S., Banu, R., Hamzah, H., … Wong, T. Y. (2020). A deep learning algorithm to detect chronic kidney disease from retinal photographs in community-based populations. The Lancet Digital Health, 2(6). https://doi.org/10.1016/S2589-7500(20)30063-7
22. Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.
23. Mikołajczyk, A., & Grochowski, M. (2018). Data augmentation for improving deep learning in

image classification problem. *2018 International Interdisciplinary PhD Workshop, IIPhDW 2018*. https://doi.org/10.1109/IIPHDW.2018.8388338

24. Rawat, W., & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*. https://doi.org/10.1162/NECO_a_00990
25. Wang, S., Su, J. H., Beliveau, B. J., Bintu, B., Moffitt, J. R., Wu, C. T., & Zhuang, X. (2016). Spatial organization of chromatin domains and compartments in single chromosomes. Science, 353(6299). https://doi.org/10.1126/science.aaf8084
26. Sawh, A. N., Shafer, M. E. R., Su, J. H., Zhuang, X., Wang, S., & Mango, S. E. (2020). Lamina-Dependent Stretching and Unconventional Chromosome Compartments in Early C. elegans Embryos. Molecular Cell, 78(1). https://doi.org/10.1016/j.molcel.2020.02.006
27. Rosin, L. F., Nguyen, S. C., & Joyce, E. F. (2018). Condensin II drives large-scale folding and spatial partitioning of interphase chromosomes in Drosophila nuclei. PLoS Genetics, 14(7). https://doi.org/10.1371/journal.pgen.1007393
28. Fields, B. D., Nguyen, S. C., Nir, G., & Kennedy, S. (2019). A multiplexed dna fish strategy for assessing genome architecture in caenorhabditis elegans. ELife, 8. https://doi.org/10.7554/eLife.42823
29. Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., … Cardona, A. (2012). Fiji: An open-source platform for biological-image analysis. *Nature Methods*. https://doi.org/10.1038/nmeth.2019
30. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science and Engineering*, *9*(3). https://doi.org/10.1109/MCSE.2007.55.
31. Hashemi, M. (2019). Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0263-7
32. Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0197-0
33. Shen, D., Wu, G., & Suk, H. Il. (2017). Deep Learning in Medical Image Analysis. Annual Review of Biomedical Engineering, 19. https://doi.org/10.1146/annurev-bioeng-071516-044442
34. Lee, J. H., Daugharthy, E. R., Scheiman, J., Kalhor, R., Yang, J. L., Ferrante, T. C., … Church, G. M. (2014). Highly multiplexed subcellular RNA sequencing in situ. Science, 343(6177). https://doi.org/10.1126/science.1250212
35. Schmidhuber, J. (2015). Deep Learning in neural networks: An overview. *Neural Networks*. https://doi.org/10.1016/j.neunet.2014.09.003
36. Goh, T. Y., Basah, S. N., Yazid, H., Aziz Safar, M. J., & Ahmad Saad, F. S. (2018). Performance analysis of image thresholding: Otsu technique. *Measurement: Journal of the International Measurement Confederation*, *114*. https://doi.org/10.1016/j.measurement.2017.09.052
37. Singh, S. P., Wang, L., Gupta, S., Goli, H., Padmanabhan, P., & Gulyás, B. (2020, September 2). 3d deep learning on medical images: A review. *Sensors (Switzerland)*. MDPI AG. https://doi.org/10.3390/s20185097
38. Sudeep, K. S., & Pal, K. K. (2017). Preprocessing for image classification by convolutional neural networks. In *2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings*. https://doi.org/10.1109/RTEICT.2016.7808140
39. Janssens, A. C. J. W., & Martens, F. K. (2020). Reflection on modern methods: Revisiting the area under the ROC Curve. *International Journal of Epidemiology*, *49*(4). https://doi.org/10.1093/ije/dyz274
40. Rawat, W., & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*. https://doi.org/10.1162/NECO_a_00990