

# Chromosomes Recognition

Matteo Tinacci e Leonardo Pampaloni

28/05/2024



UNIVERSITÀ  
DEGLI STUDI  
FIRENZE

# Table of Contents

## 1 Introduction

► Introduction

► First Shot

► Second Approach

► Conclusions

# Integrating AI into Genome Analysis Software

## 1 Introduction

- **Project Background:**
  - Original software: Chromosome recognition based on image similarities
  - Lacked AI for enhanced recognition
  - Aimed to meet contract requirements for deep learning integration
- **Motivation:**
  - Improve accuracy beyond the existing 70% score
  - Achieve reliable automation without human supervision
- **Objective:**
  - Fully automate chromosome recognition using AI
  - Enhance efficiency and reliability in genome analysis

# Table of Contents

## 2 First Shot

► Introduction

► **First Shot**

► Second Approach

► Conclusions

# Dataset Structure

## 2 First Shot

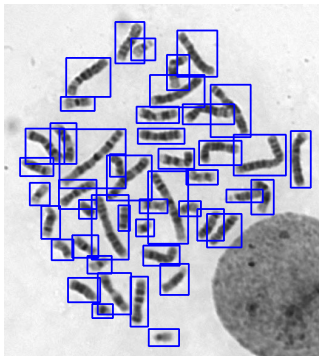
- The dataset consists of two folders:
  - Images folder: Contains a series of images
  - Annotations folder: Contains XML files representing annotations for each corresponding image
- Each image contains all 24 chromosomes along with noise and artifacts



# Data Refinement

## 2 First Shot

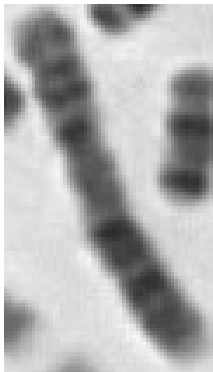
- Annotations provided for each image:
  - Class of each chromosome
  - Position of each chromosome
- Simplified the cropping of each class from each image



# Data Segmentation

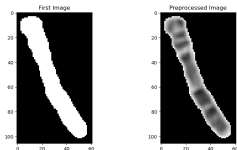
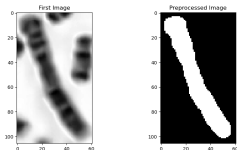
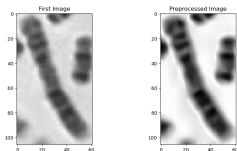
## 2 First Shot

- Created subfolders containing chromosomes associated with each respective class
- Extracted from the original karyotype image



# Preprocessing

## 2 First Shot

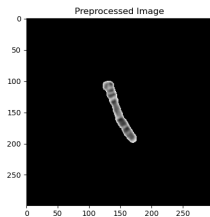
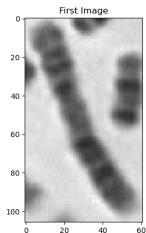


- Cropping: Removing unwanted parts of the image.
- Noise Reduction: Smoothing out the image to reduce noise.
- Contrast Enhancement: Improving the contrast of the image.
- Largest Component: Extracting the largest object in the image.
- Morphological Operations: Erosion, watershed algorithm.
- Applied Mask: Using masks to highlight specific regions.



# Final result

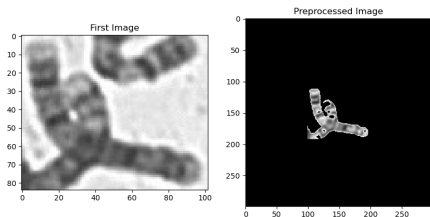
## 2 First Shot



After preprocessing, resizing and cropping  
this is the final result

# The problem

## 2 First Shot



- The dataset consists of poor quality images, exemplified by the one shown here.
- This low quality prompted us to seek higher quality alternatives.
- The primary issue with the dataset was image overlapping, which significantly affected our results.

# Table of Contents

## 3 Second Approach

► Introduction

► First Shot

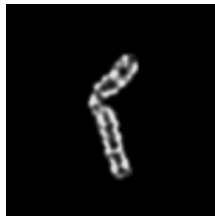
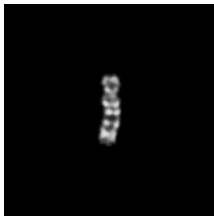
► **Second Approach**

► Conclusions

# Dataset Overview

## 3 Second Approach

The dataset contains 130 noise-free images for each type of chromosome with black backgrounds. Each image is 224x224 pixels in TIFF format, with the classification indicated in the filename.



# Model Selection

## 3 Second Approach

Initially, we attempted to implement the InceptionResNet v2 model, but faced compatibility issues with TensorFlow. Hence, we explored alternative models like InceptionV3, ResNet50, and ResNet18 in PyTorch.

# Data Augmentation

## 3 Second Approach

D'Angelo and Nanni proposed augmentation algorithms for effective model training, achieving an area under the ROC curve of 0.9999. They include augmentation and straightening.



(a)



(b)

# Preprocessing

## 3 Second Approach

Regarding model training, we adapted the dataset by applying padding and resizing to meet the requirements of each model. For example, InceptionV3 requires images of size 299x299. We then applied augmentation, including:

- Contrast, brightness, saturation, and intensity variation to vary chromatic characteristics.
- Rotation, translation, and scaling to change position, orientation, and size.
- Gaussian blur with variable kernel sizes (5x5 to 9x9) and standard deviations (0.1 to 0.5).

Finally, we randomly split the dataset into Train, Test, and Validation sets, with portions of 70%, 10%, and 20%, respectively.

# Hyperparameters

## 3 Second Approach

Regarding model training hyperparameters:

- **Optimizer:** Tested Adam, SGD, and RMSprop. Adam combines momentum and RMSprop benefits. SGD updates parameters by computing gradients on small batches of data. RMSprop adjusts learning rate based on recent gradients, effective for sparse gradients.
- **Learning rate:** Tested 0.0001, 0.001, 0.01.

Batch size: 16, maximum supported by GPU.



# Hyperparameters

## 3 Second Approach

After plotting grid search graphs and observing discontinuous results, implemented:

- Learning rate scheduler: Decreases rate by 0.1 every 15 epochs.
- Weight decay: Prevents overfitting by penalizing large weights, applied decay of  $1e-5$ .

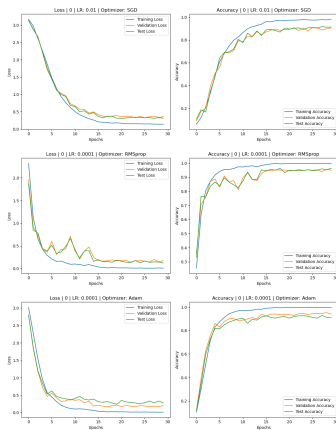
Implemented early stop system after observing overfitting tendencies, stopping training when accuracy stabilizes for 10 epochs.

The first model considered was InceptionV3.

- InceptionV3: A deep CNN architecture designed for image classification and feature extraction. It utilizes sophisticated "Inception" modules to efficiently capture features at different scales.
- Achieves state-of-the-art performance on image classification benchmarks and widely used in computer vision applications.
- Implemented starting from pre-trained model in PyTorch library, modifying last layer for 24 output classes, one for each chromosome.

# Grid Search for InceptionV3

## 3 Second Approach



As we can see from the graphs, the best optimizers for InceptionV3 are SGD and Adam. For SGD, the best result is achieved with a higher learning rate, while for Adam, a very low learning rate is optimal. RMSprop also performs decently with a low learning rate but exhibits spikes during validation and test phases not present in the other two optimizers, making it the worst of the three.

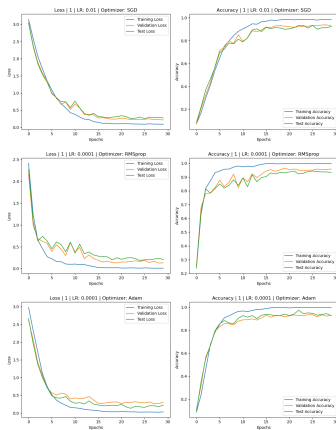
# Alternative InceptionV3

## 3 Second Approach

While searching online, we came across a paper proposing an alternative version of InceptionV3 with modified fully connected layers, still aimed at chromosome recognition. As seen from the graphs, this alternative version performs well with different optimizers and learning rates, indicating its potential for chromosome recognition tasks.

# Grid Search for Alternative InceptionV3

## 3 Second Approach



Observing the graphs, we can notice that the behavior is rather similar to that of the original model. We do not observe a substantial improvement or worsening but only a lower height of the spikes when using RMSprop with a learning rate of 0.0001 and a slight improvement in accuracy for Adam with the same learning rate.

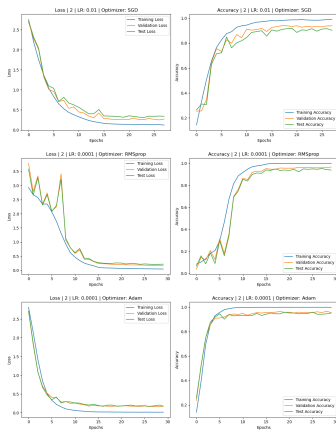
# VGG16

## 3 Second Approach

VGG16 is a convolutional neural network (CNN) architecture renowned for its simplicity and effectiveness in image classification tasks. It consists of 16 layers, including convolutional layers with small 3x3 filters and max-pooling layers, resulting in a deep network capable of learning rich hierarchical features. VGG16 has been widely used as a baseline model in various computer vision applications and is known for its strong performance on standard image classification benchmarks.

# Grid Search for VGG16

## 3 Second Approach



Also in this case, we can notice that the best among the 3 optimizers is Adam with a learning rate of 0.0001, particularly performing slightly better than the previous InceptionV3 models. SGD is good for high learning rates while worsening as the learning rate decreases, but not as much as for the previous InceptionV3 models. RMSprop is a valid optimizer for very low learning rates but cannot keep up with Adam.

# ResNet50

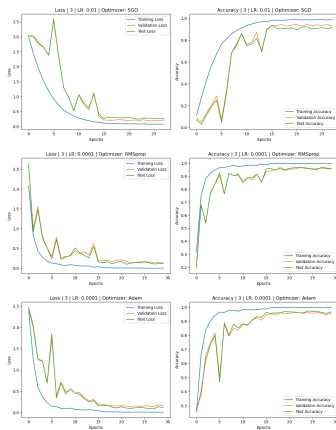
## 3 Second Approach

ResNet50 is a deep convolutional neural network (CNN) with 50 layers, notable for its introduction of residual learning. Its architecture includes residual blocks with skip connections, enabling effective training of very deep networks by addressing the vanishing gradient problem. With pre-trained weights available, it's widely used in various computer vision tasks like image classification, object detection, and segmentation.



# Grid Search for Resnet50

## 3 Second Approach



Observing the graphs, RMSprop emerges as the most successful optimizer for ResNet50, particularly performing well for very low learning rates. SGD achieves low loss for high learning rates but exhibits high spikes, while Adam's results are notably worse compared to previous models. Due to the depth of the network and potential overfitting, we opted to explore ResNet18 as a lighter alternative.

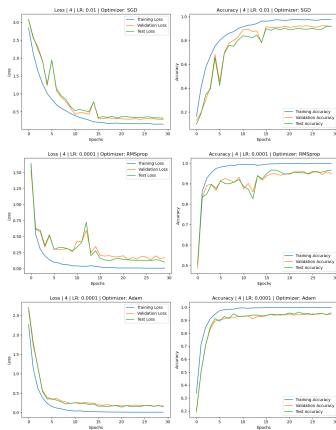
# ResNet18

## 3 Second Approach

ResNet18 is a lighter variant of ResNet with 18 layers, designed for scenarios with limited computational resources. Despite its simpler architecture, it offers strong performance in tasks like image classification. It's favored for applications where memory and computation resources are constrained, such as mobile devices and real-time systems.

# Grid Search for Resnet18

## 3 Second Approach



Adam optimizer performs excellently for ResNet18, comparable to VGG16 and superior to InceptionV3 and ResNet50. Other optimizers are effective but not as good as Adam with a learning rate of 0.0001. There are indications of potential overfitting in ResNet50, as its loss function improves suddenly after about 20 epochs, unlike ResNet18, which maintains consistently low loss.

# Model Performance Comparison

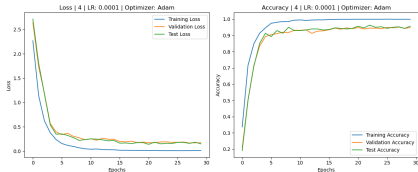
## 3 Second Approach

Model	Learning Rate (lr)	Optimizer	Validation Accuracy
V3Inception	0.001	Adam	96.0%
V3InceptionPaper 2	0.0001	RMSprop	95.8%
VGG16	0.0001	Adam	96.8%
Resnet50	0.01	SGD	99.0%
Renset18	0.0001	Adam	96.2%

**Table:** Best Hyperparameters for each model

# Conclusion

## 3 Second Approach

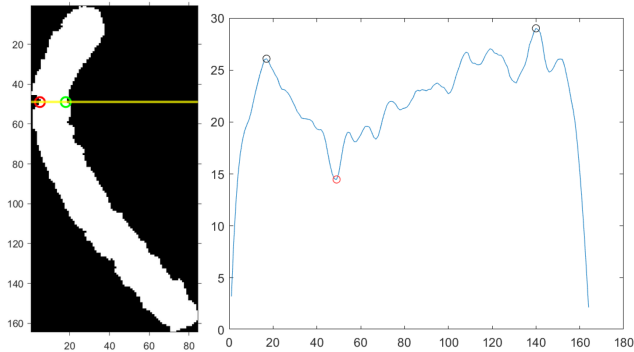


Several models performed well, but ResNet18 shows faster training. Additionally, it was our choice due to being the lightest network among those tested, weighing 44 MB, with shorter training times and better integration capabilities into external systems.

# State of Art

## 3 Second Approach

The current state of the art includes techniques for straightening the chromosome through centromere identification. Currently, there are methods for reconstructing the missing part of the overlapping chromosome but they have not been explored.

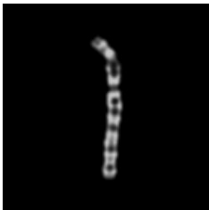


### Image Prediction

0156282825...9057.5.jpeg

#### Prediction:

5



Created a Flask-based app utilizing trained models with optimized hyperparameters. The app allows image classification via API calls. An HTML interface enables easy image upload and classification retrieval.

# Table of Contents

## 4 Conclusions

► Introduction

► First Shot

► Second Approach

► Conclusions



# Conclusion

## 4 Conclusions

The system demonstrates an impressive prediction accuracy of **99%** (Resnet50) on a properly preprocessed dataset. However, there is a need to explore more sophisticated methods for dataset preprocessing.

Possible future approaches include:

- Chromosome straightening to reduce overlapping.
- Reconstruction of overlapping chromosomes.
- Use of Segment-Anything-Model (SAM) for chromosome segmentation.

# Chromosomes Recognition

*Thank you for listening!*  
*Any questions?*