Machine learning applied to population genomics

Manolo Fernandez Perez
Departent of Life Sciences, Imperial College London

@ManoloLearning
manolofperez@gmail.com
sites.google.com/site/manolofperez

Goals

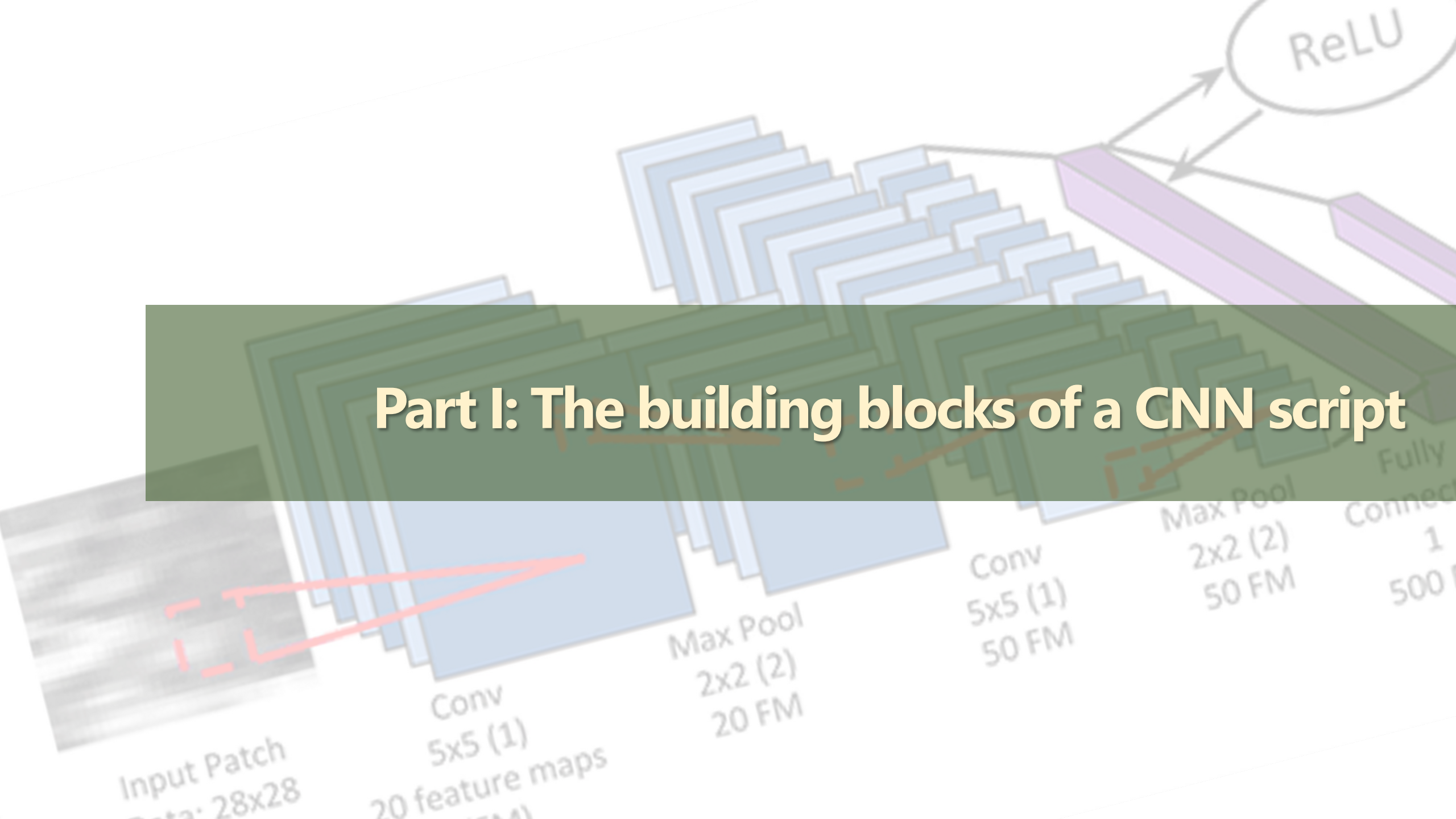
- Conceive and simulate genetic data under competing demographic scenarios 

- Understand deep learning background and how a CNN works 

- Use CNN to detect regions with selective sweeps on real genomes

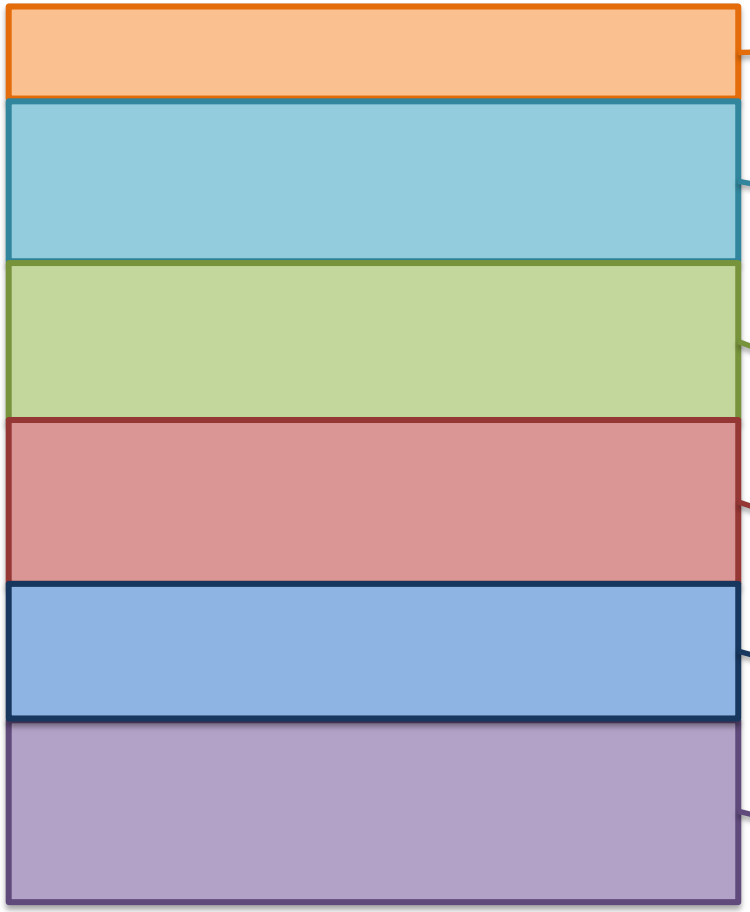
- How to use deep learning to compare demographic scenarios

Program



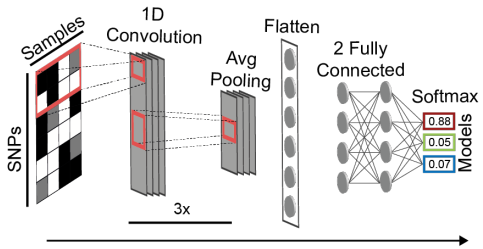
Part I: The building blocks of a CNN script

CNN Script



required python modules.

Define the CNN architecture.



Load and process the training data.

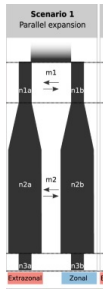
Train the network.

Load the test data and perform cross-validation.

Predict the most likely model for the empirical data

CNN Script

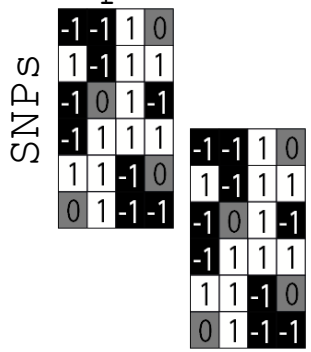
Inputs:

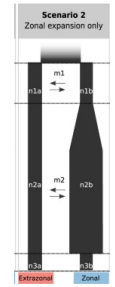


Scenario 1

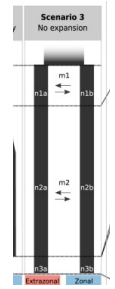
Samples

SNPs

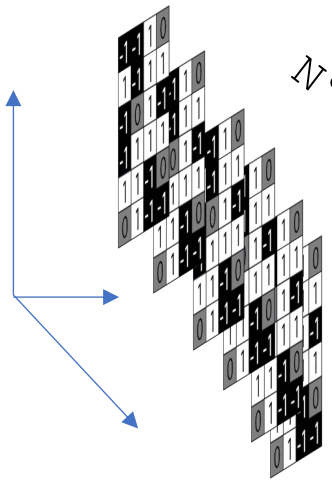




Scenario 2



Scenario 3



N° of simulations

3-D Numpy array

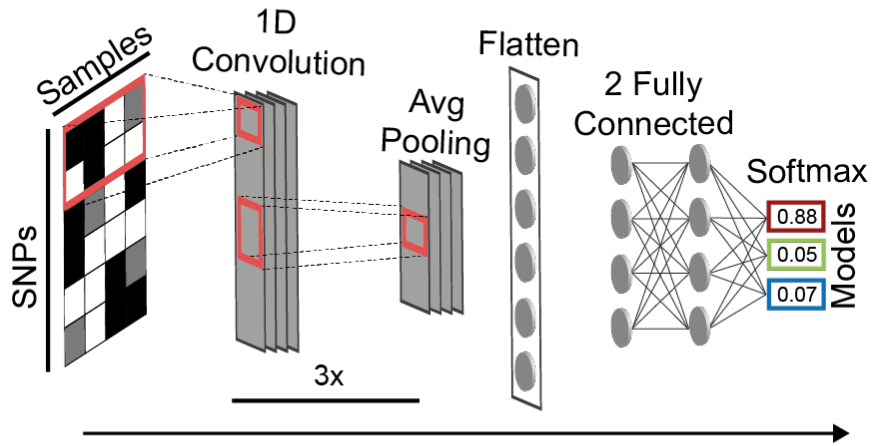
Parameters

	Theta	T1	T2	T3	Ne
Sim1					
Sim2					
Sim3					
Sim4					

- **Practical Exercise 1:**

- Go through Section 1 of the Part1 script (Demographic models) and try to recognize all the elements of the network. Do you remember the function of each of those elements? Remember that you can add annotations to the code using # and add information that might help you when you get back to the script in the future.

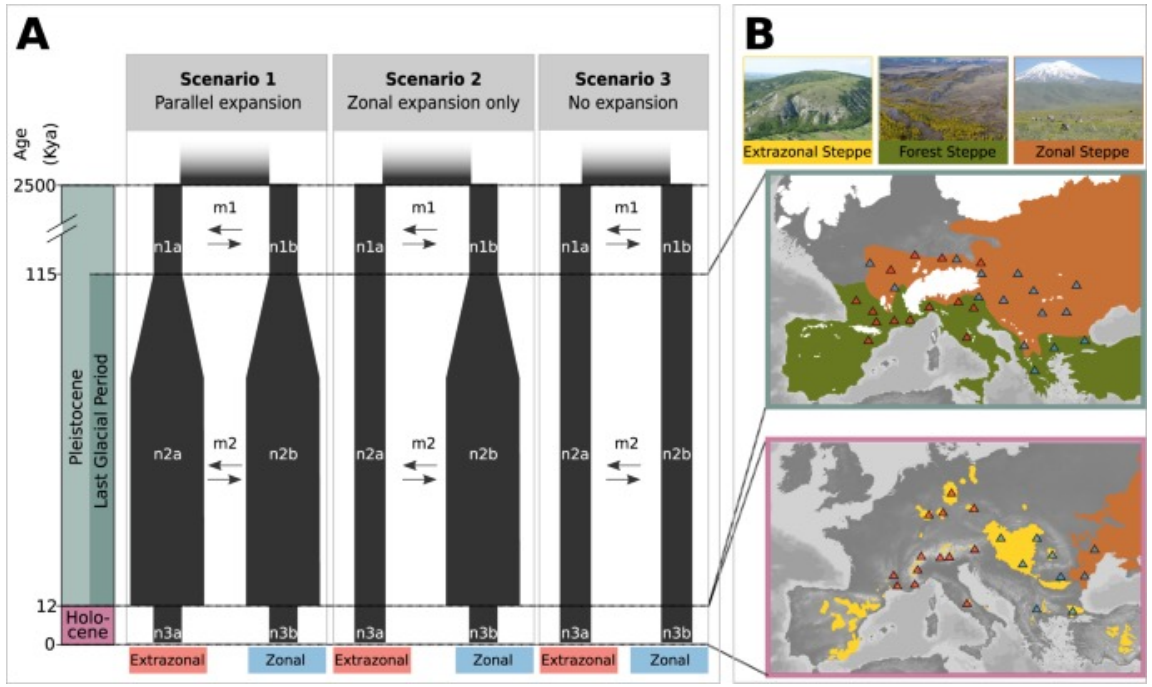
- Now run all the cells until you reach the end of section 2. Your network will be training, so now we will have some time to discuss and do a quick review on the CNN elements.

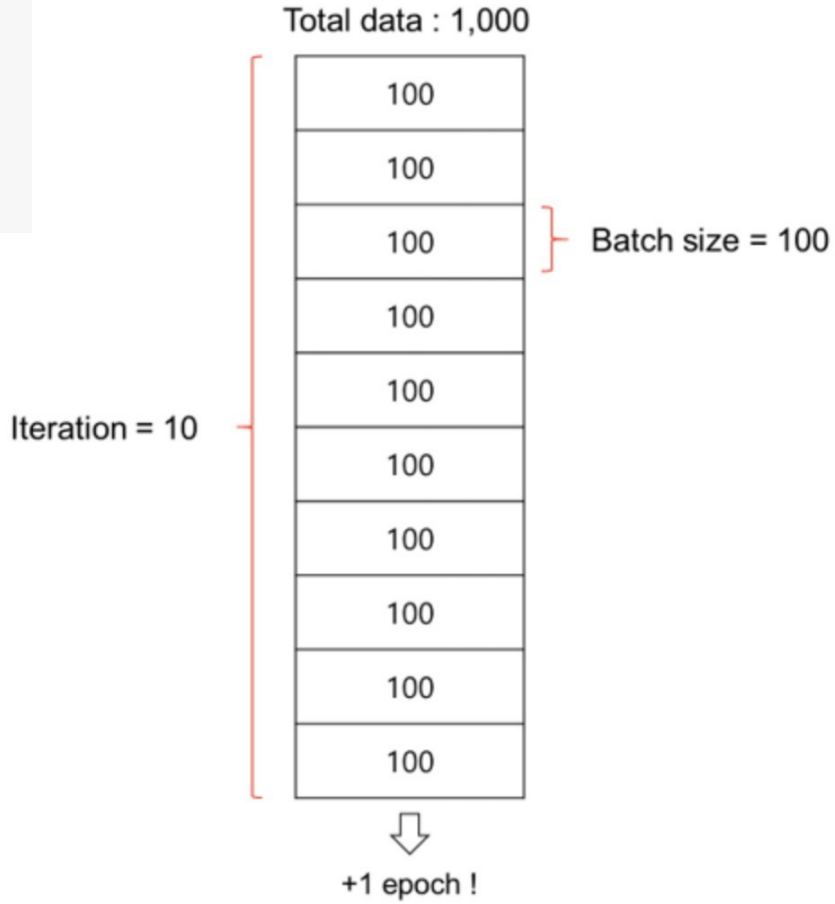


CNN Script

```
# Define parameters for the CNN run.
batch_size = 200
### how much interations to train the network
epochs = 100

###n of models
num_classes = 3
```



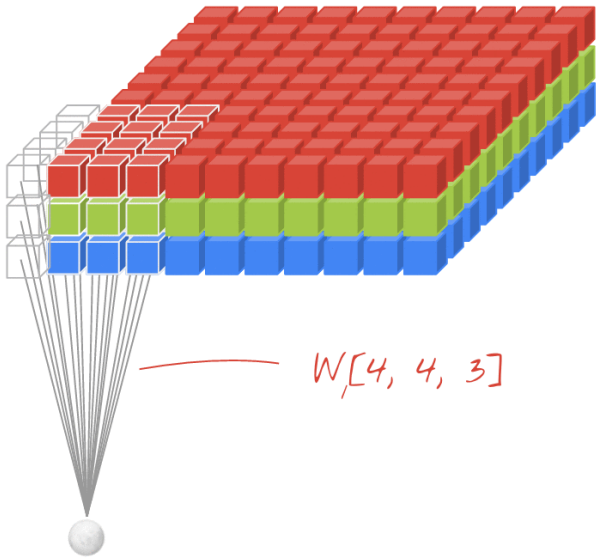


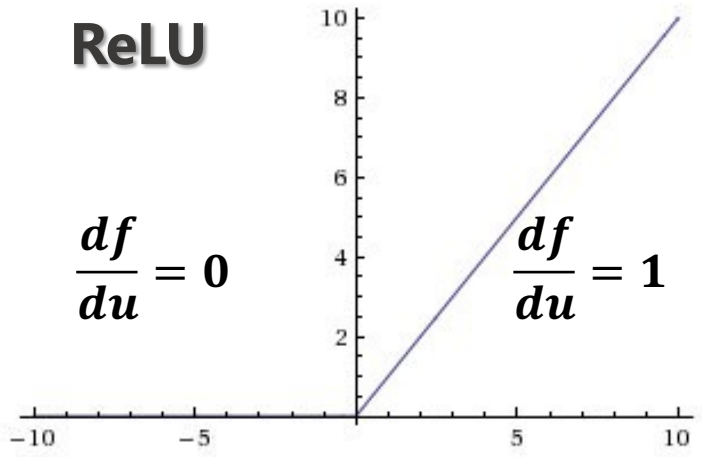
https://jerryan.medium.com/batch-size-a15958708a6

Kirschner et al. (2022) *Nat Comm*

CNN Script

```
# Define the CNN architecture.
def create_cnn(xtest):
  inputShape = (xtest.shape[1], xtest.shape[2])
  ## image size. images need to have EXACTLY the same size
  inputs = Input(shape=inputShape)
  x = inputs
  ## 1D convolution - less computational intensive and also treats snps as independent;
  x = Conv1D(250, kernel_size=2, activation='relu',input_shape=(xtest.shape[1], xtest.shape[2]))(x)
  ### Enables the network to learn more complex features / shapes.
  x = AveragePooling1D(pool_size=2)(x)
  x = BatchNormalization()(x)
```



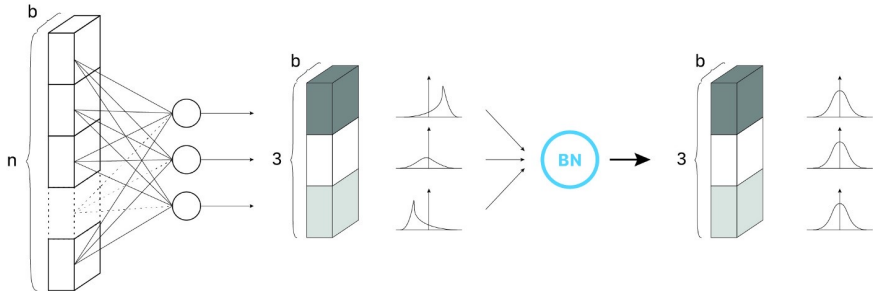


https://www.kaggle.com/code/dansbecker/rectified-linear-units-relu-in-deep-learning/notebook

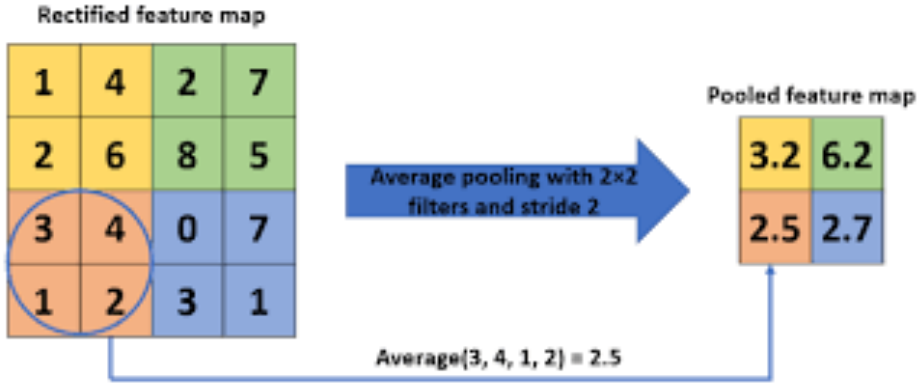
Kirschner et al. (2022) *Nat Comm*

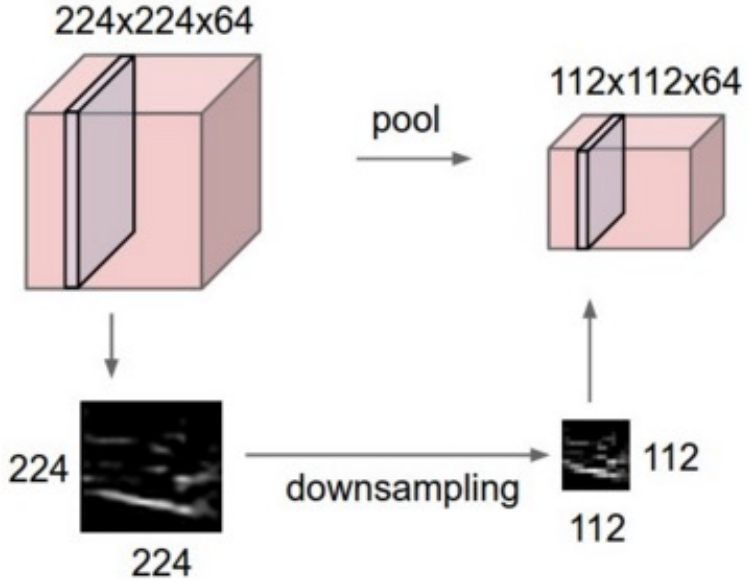
CNN Script

```
# Define the CNN architecture.
def create_cnn(xtest):
  inputShape = (xtest.shape[1], xtest.shape[2])
  ## image size. images need to have EXACTLY the same size
  inputs = Input(shape=inputShape)
  x = inputs
  ## 1D convolution - less computational intensive and also treats snps as independent;
  x = Conv1D(250, kernel_size=2, activation='relu',input_shape=(xtest.shape[1], xtest.shape[2]))(x)
  ### Enables the network to learn more complex features / shapes.
  x = AveragePooling1D(pool_size=2)(x)
  x = BatchNormalization()(x)
```



https://towardsdatascience.com/batch-normalization-in-3-levels-of-understanding-14c2da90a338





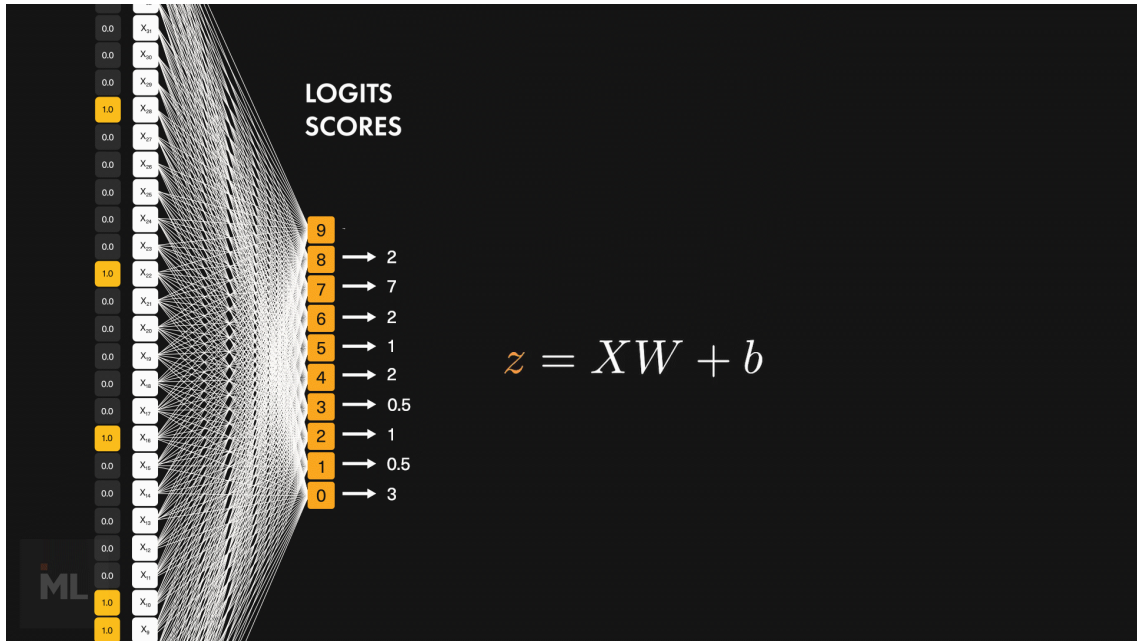
https://leonardoaraujosantos.gitbook.io/artificial-inteligence/machine_learning/deep_learning/pooling_layer

Gholamalinezhad & Khosravi (2020) *arXiv*

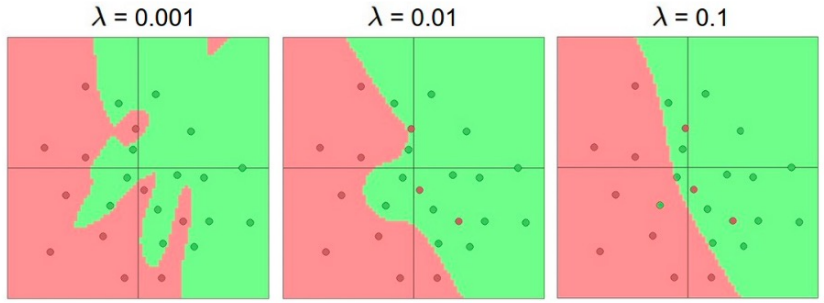
Kirschner et al. (2022) *Nat Comm*

CNN Script

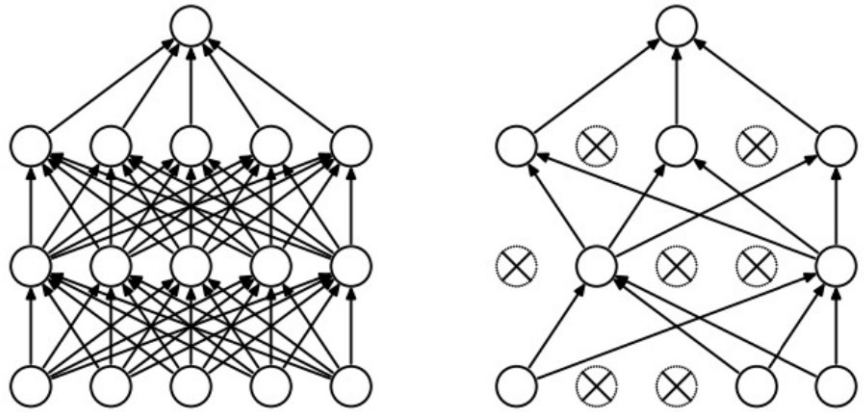
```
### Linearising the image as in the initial step. From this point on the network behaves as a Multi-L
x = Flatten()(x)
x = Dense(125, activation='relu',kernel_regularizer=l2(1e-3), bias_regularizer=l2(1e-3))(x)
x = Dropout(0.5)(x)
x = Dense(125, activation='relu',kernel_regularizer=l2(1e-3), bias_regularizer=l2(1e-3))(x)
x = Dropout(0.5)(x)
x = Dense(num_classes, activation="softmax")(x)
```



Regularization



Options: L2, L1, maxnorm and dropout.

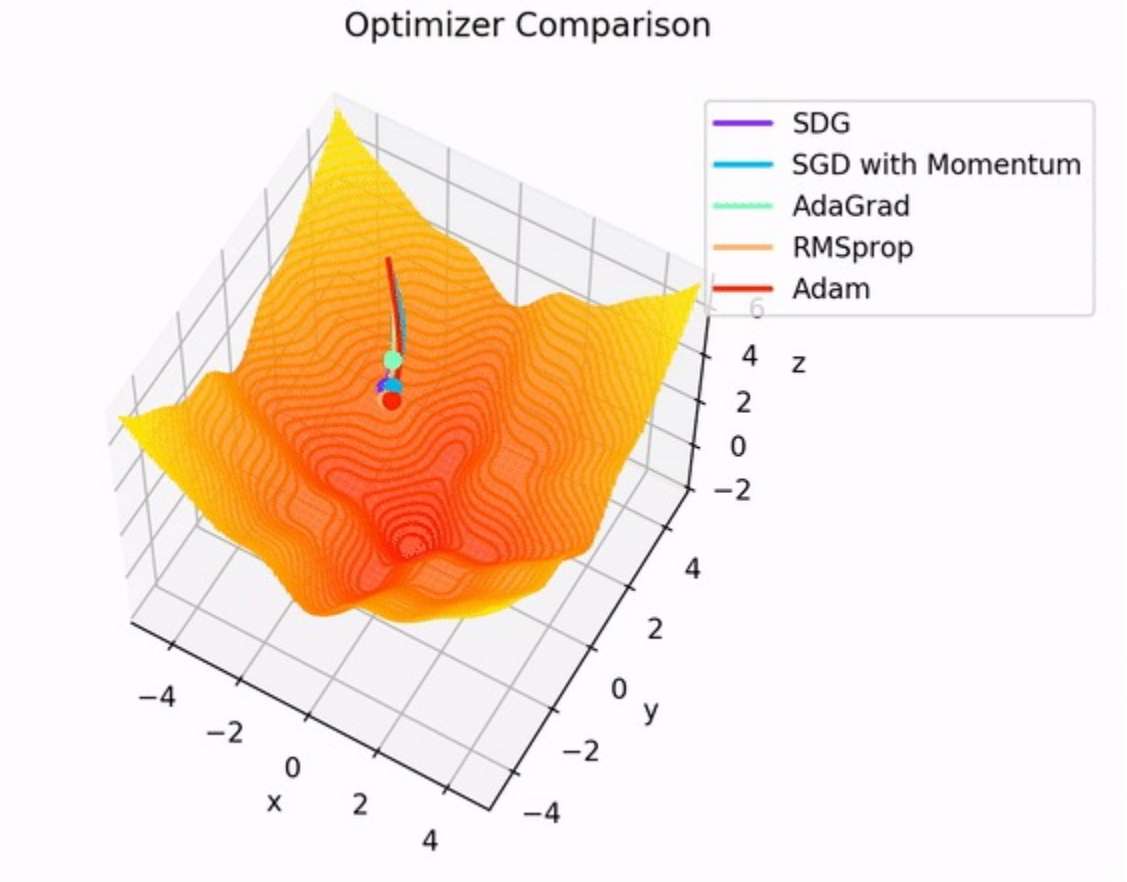


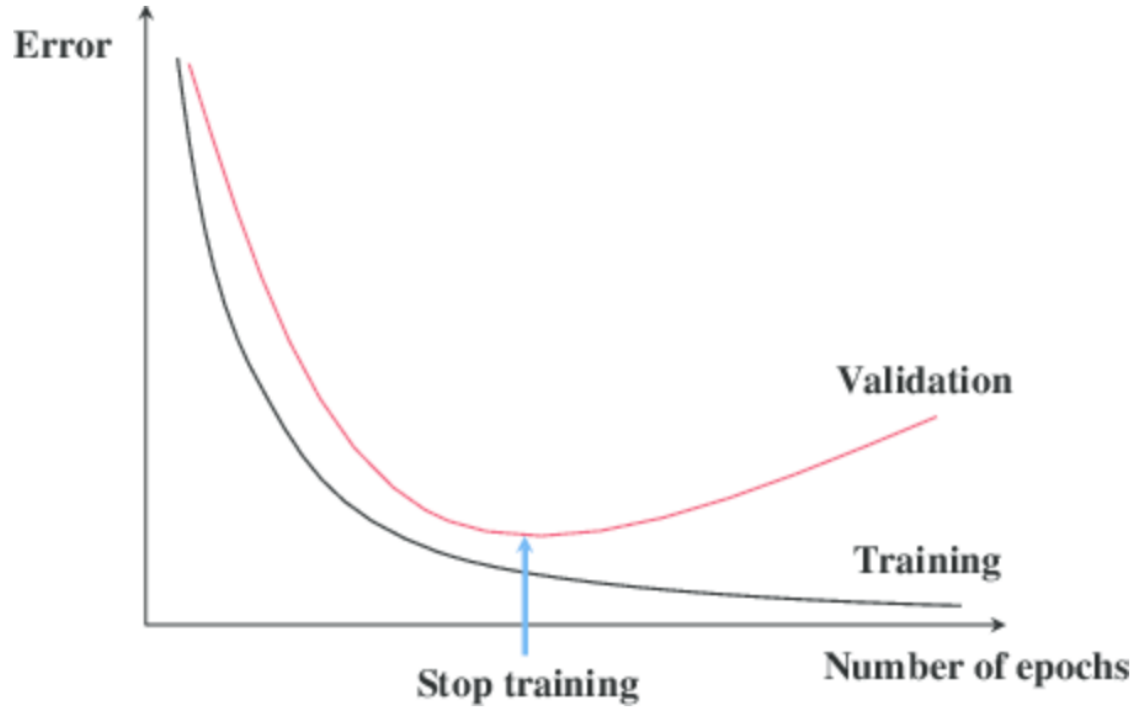
(a) Standard Neural Net (b) After applying dropout.

CNN Script

```
# Compile the CNN.
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer='Adam',
              metrics=['accuracy'])

# We will use early stopping and save the model with the best val_accuracy.
earlyStopping = EarlyStopping(monitor='val_accuracy', patience=150, verbose=0, mode='max', restore_best_weights
### stop training when validation increases error
```





https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d

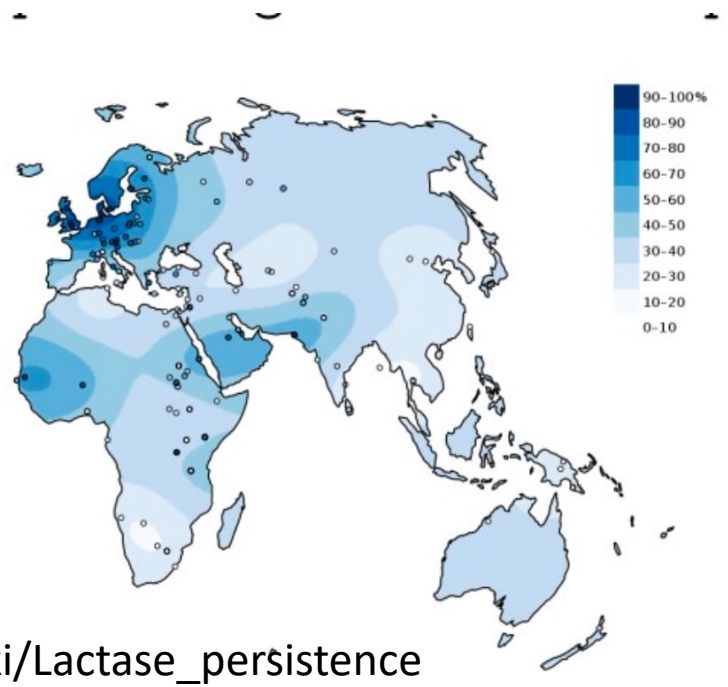
https://towardsdatascience.com/a-practical-introduction-to-early-stopping-in-machine-learning-550ac88bc8fd

- **Practical Exercise 2:**

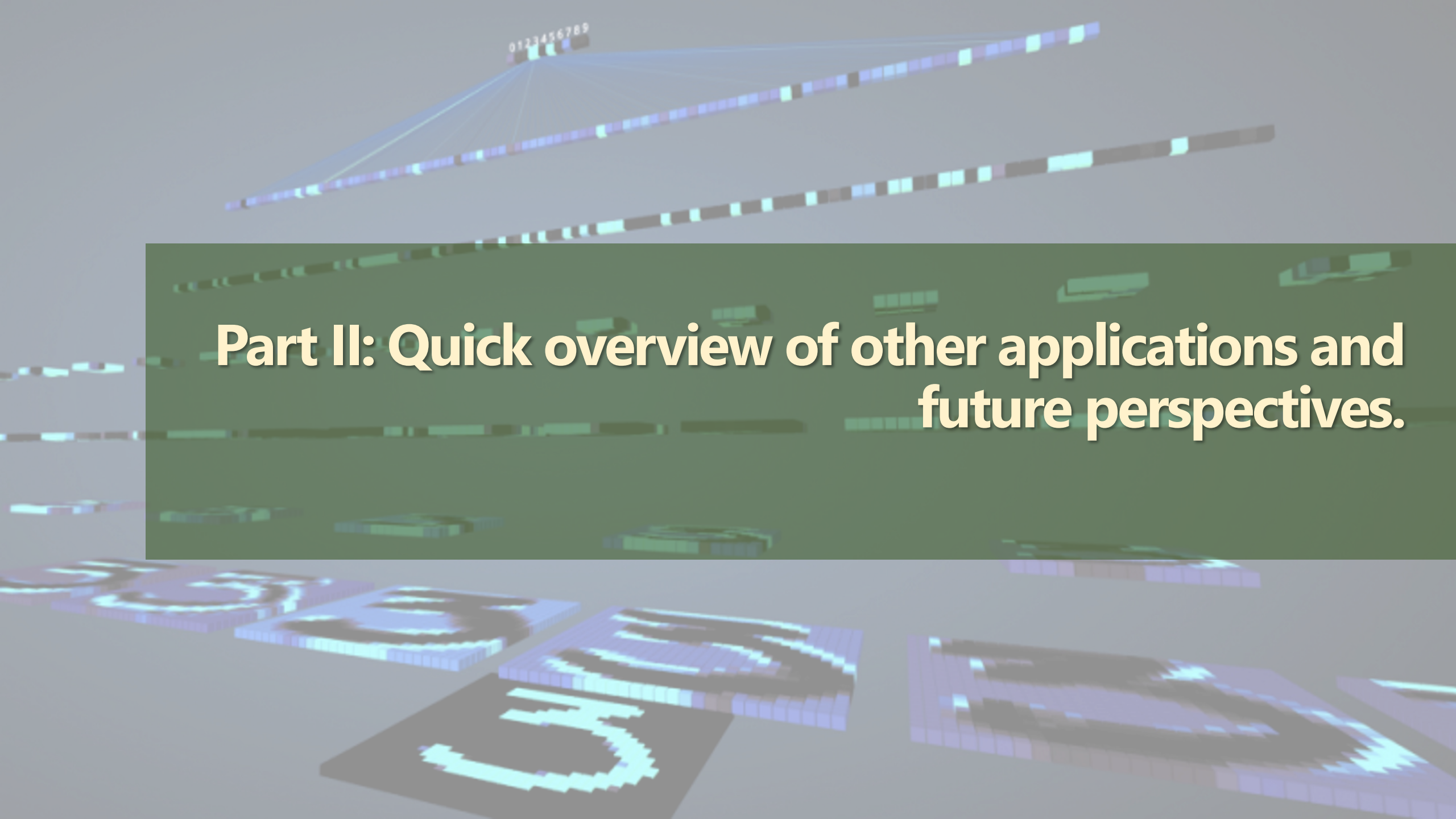
Now open the Part 2 script. It uses Matteo's software (ImaGene) that is a CNN approach to infer selection at the LCT locus.

It uses a "simulation-on-the-fly"-like approach: training data is continuously generated by simulations to avoid the network to see the same data twice and therefore to reduce overfitting. This is a valuable consideration since, when reliable simulators are available, we have access to theoretically infinite training data, the latter being constrained by computing time only.

There are also functions to automate some of the steps we did in the previous example. You can compare the strategies and the architectures adopted.



https://en.wikipedia.org/wiki/Lactase_persistence



Part II: Quick overview of other applications and future perspectives.

GBE

Deep Learning in Population Genetics

Kevin Korfmann[1], Oscar E. Gaggiotti[2], and Matteo Fumagalli [3,*]

[1]Professorship for Population Genetics, Department of Life Science Systems, Technical University of Munich, Germany
[2]Centre for Biological Diversity, Sir Harold Mitchell Building, University of St Andrews, Fife KY16 9TF, UK
[3]Department of Biological and Behavioural Sciences, Queen Mary University of London, UK

*Corresponding author: E-mail: m.fumagalli@qmul.ac.uk.

Accepted: 16 January 2023

The Unreasonable Effectiveness of Convolutional Neural Networks in Population Genetic Inference

Lex Flagel,[1,2] Yaniv Brandvain,[2] and Daniel R. Schrider*[,3]
[1]Monsanto Company, Chesterfield, MO
[2]Department of Plant and Microbial Biology, University of Minnesota, St. Paul, MN
[3]Department of Genetics, University of North Carolina, Chapel Hill, NC
***Corresponding author:** E-mail: drs@unc.edu.

Associate editor: Yuseob Kim

Review Article | Published: 04 September 2023

Harnessing deep learning for population genetic inference

Xin Huang, Aigerim Rymbekova, Olga Dolgova, Oscar Lao & Martin Kuhlwilm

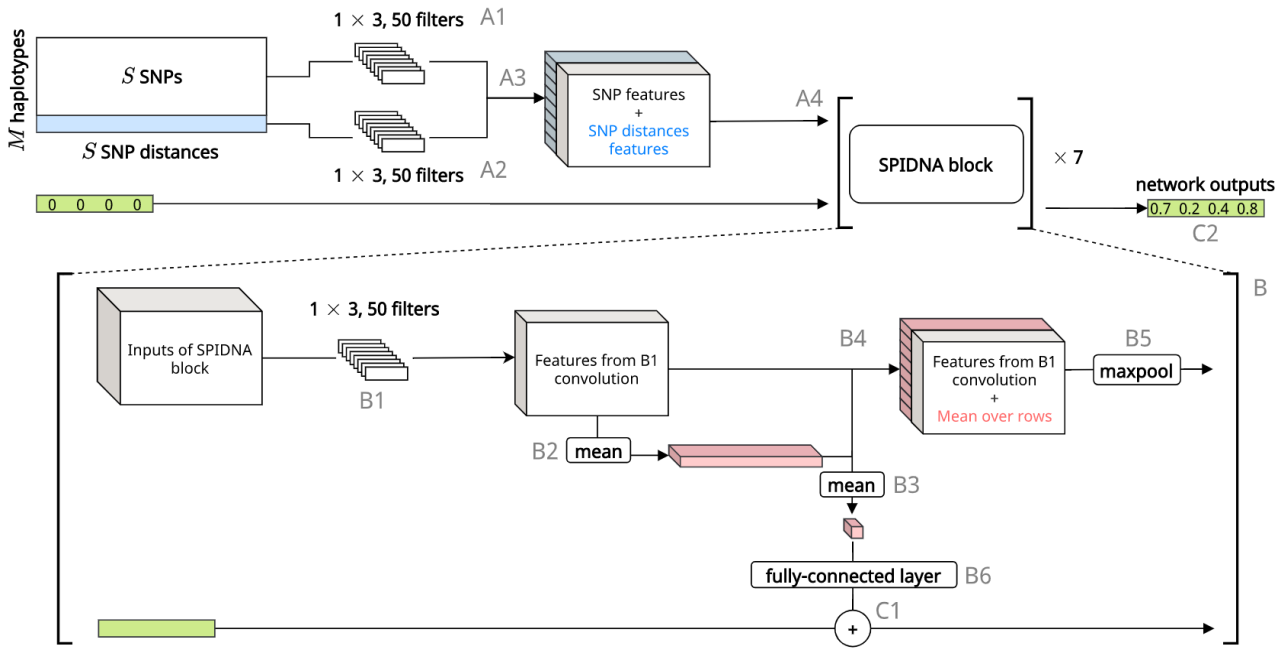
Nature Reviews Genetics **25**, 61–78 (2024) | Cite this article

SPECIAL ISSUE ARTICLE

MOLECULAR ECOLOGY RESOURCES WILEY

Deep learning for population size history inference: Design, comparison and combination with approximate Bayesian computation

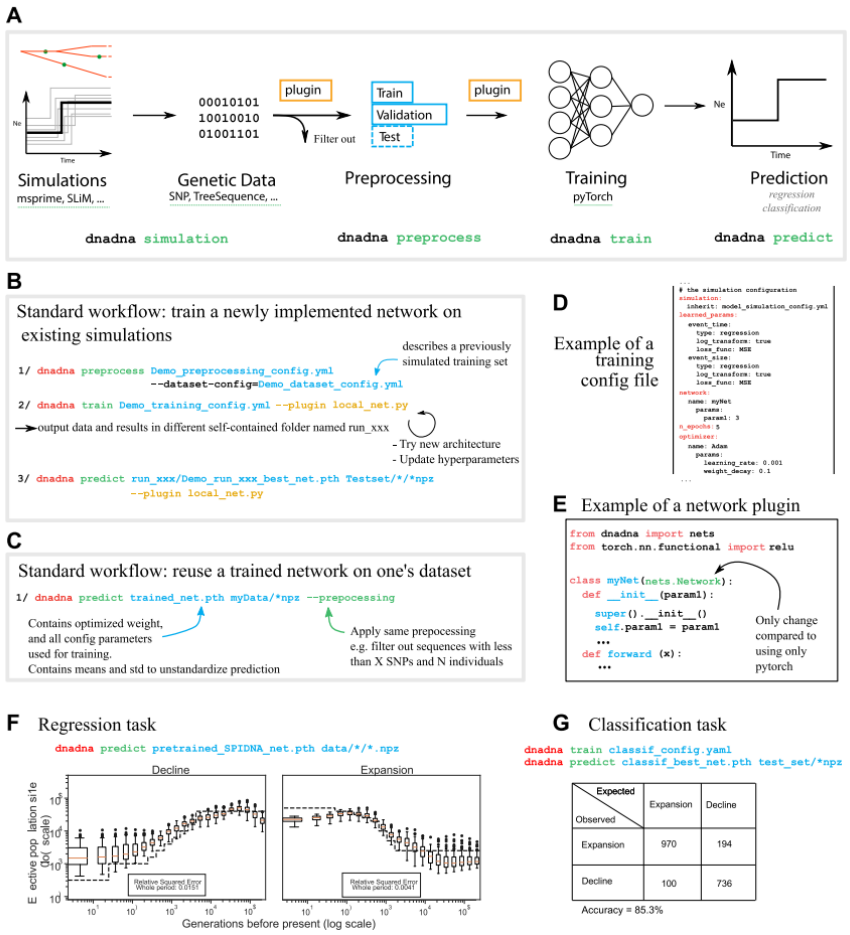
Théophile Sanchez | Jean Cury | Guillaume Charpiat | Flora Jay



Genetics and population analysis

dnadna: a deep learning framework for population genetics inference

Théophile Sanchez[1†], Erik Madison Bray[1†], Pierre Jobic[1,2], Jérémy Guez[1,3], Anne-Catherine Letournel[1], Guillaume Charpiat[1], Jean Cury[1,4*‡] and Flora Jay[1*‡]

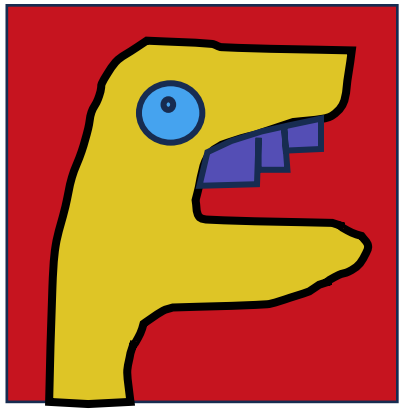


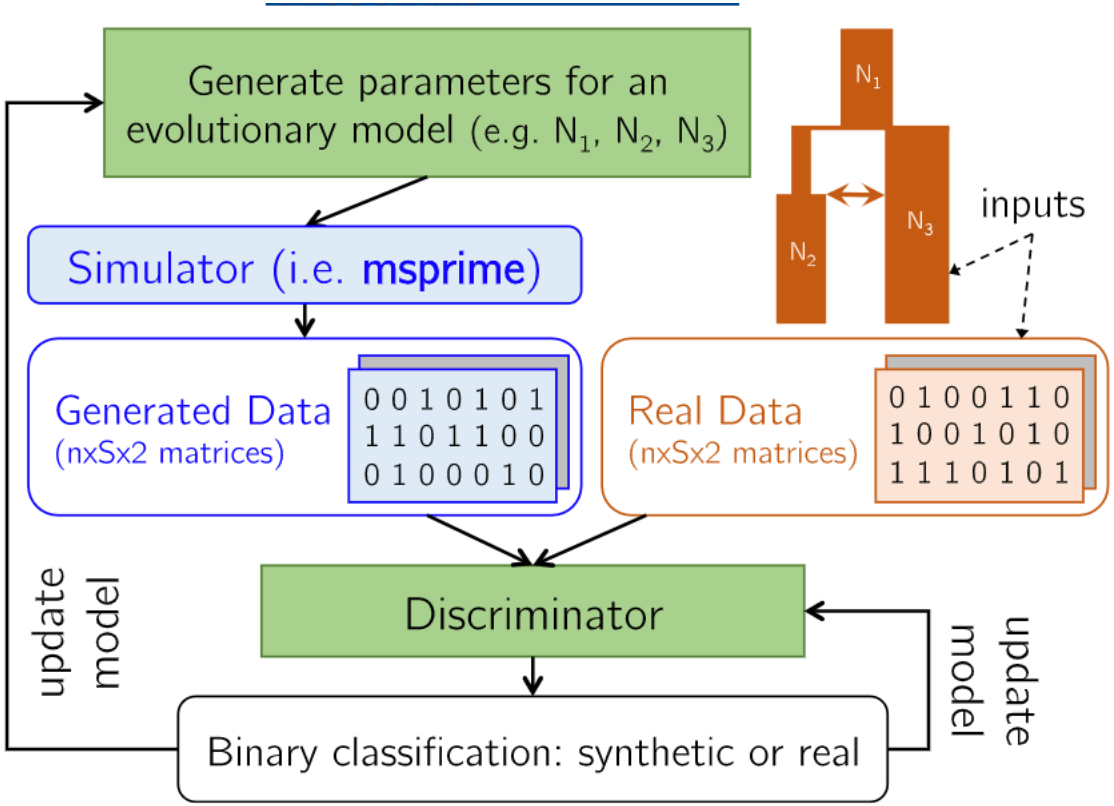
SPECIAL ISSUE

MOLECULAR ECOLOGY RESOURCES WILEY

Automatic inference of demographic parameters using generative adversarial networks

Zhanpeng Wang[1] | Jiaping Wang[1] | Michael Kourakos[2] | Nhung Hoang[2] | Hyong Hark Lee[2] | Iain Mathieson[3] | Sara Mathieson[1]



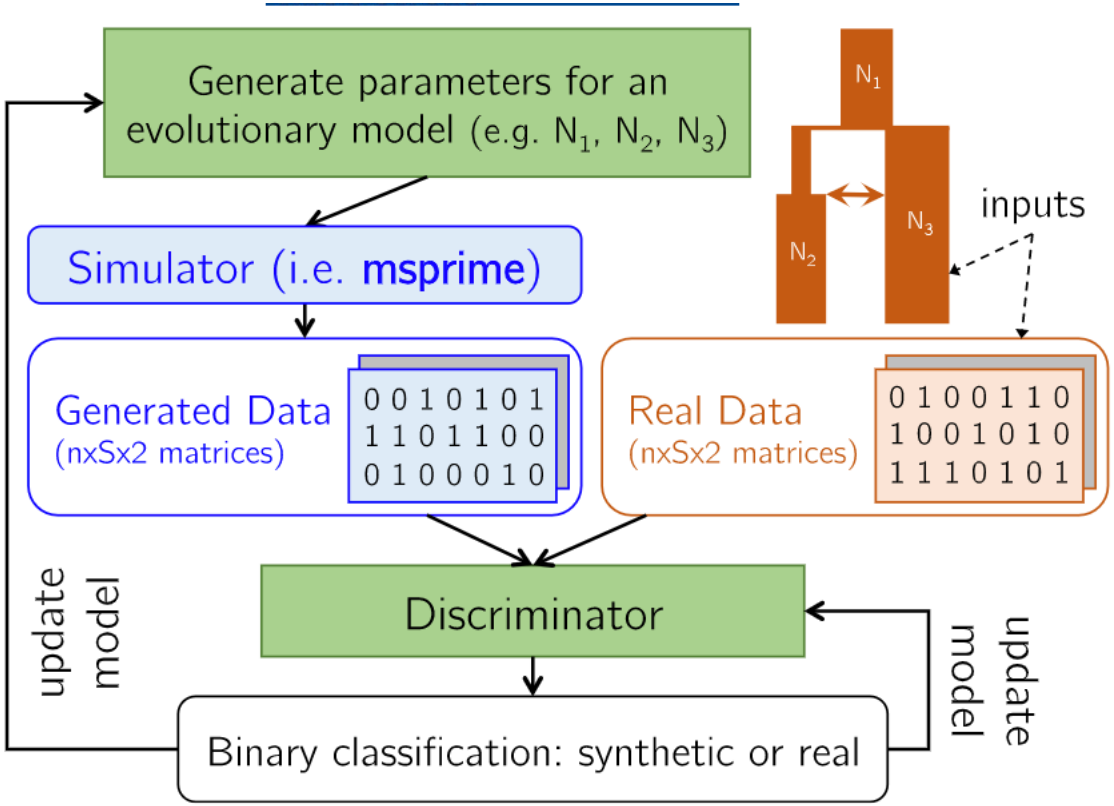


SPECIAL ISSUE

MOLECULAR ECOLOGY RESOURCES WILEY

Automatic inference of demographic parameters using generative adversarial networks

Zhanpeng Wang[1] | Jiaping Wang[1] | Michael Kourakos[2] | Nhung Hoang[2] | Hyong Hark Lee[2] | Iain Mathieson[3] | Sara Mathieson[1]



Peer Community Journal

Section: Evolutionary Biology

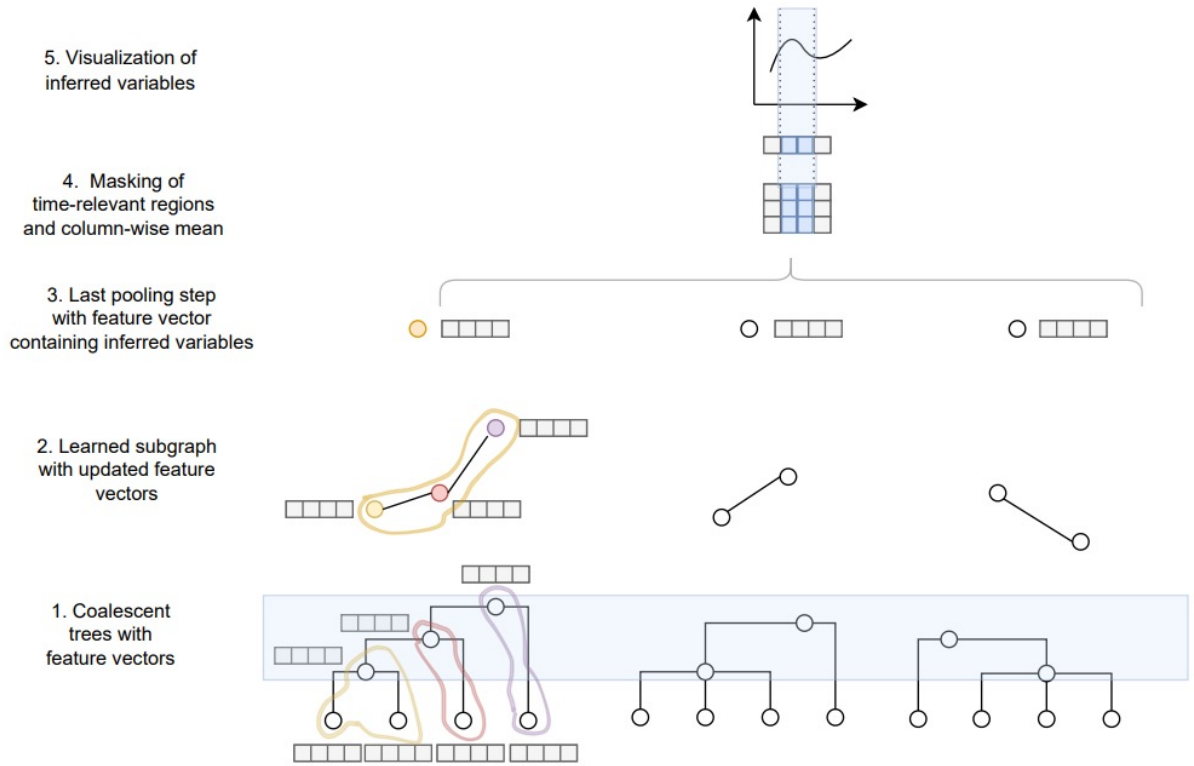
Research article

Published
2024-03-18

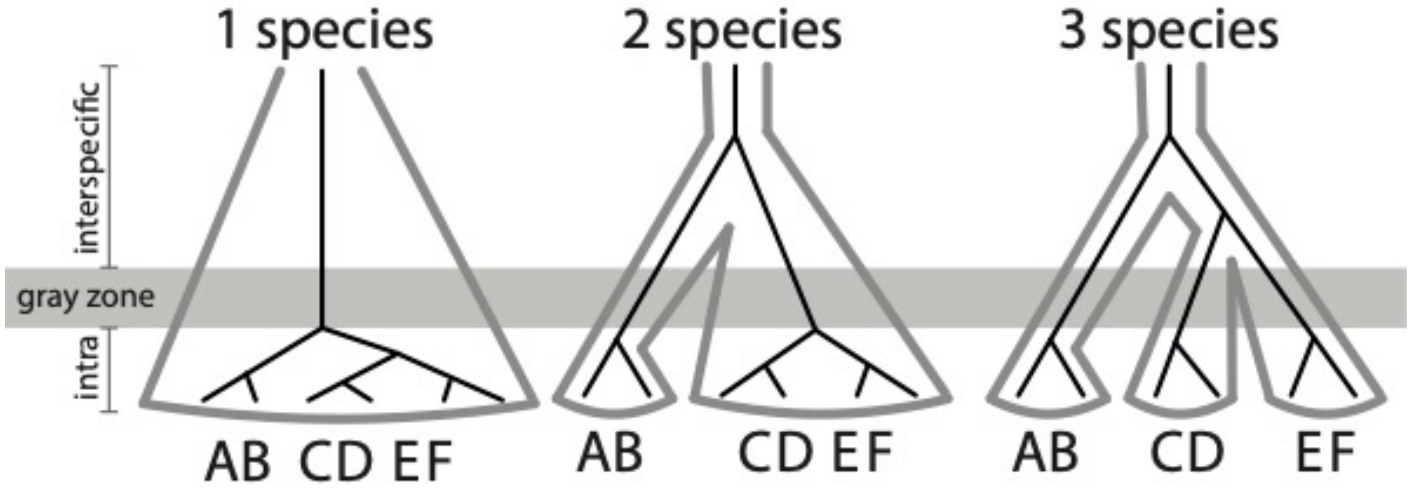
Cite as
Kevin Korfmann, Thibaut Paul Patrick Sellinger, Fabian Freund, Matteo Fumagalli and Aurélien Tellier (2024) *Simultaneous Inference of Past Demography and Selection from the Ancestral Recombination Graph under the Beta Coalescent*, Peer Community Journal, 4: e33.

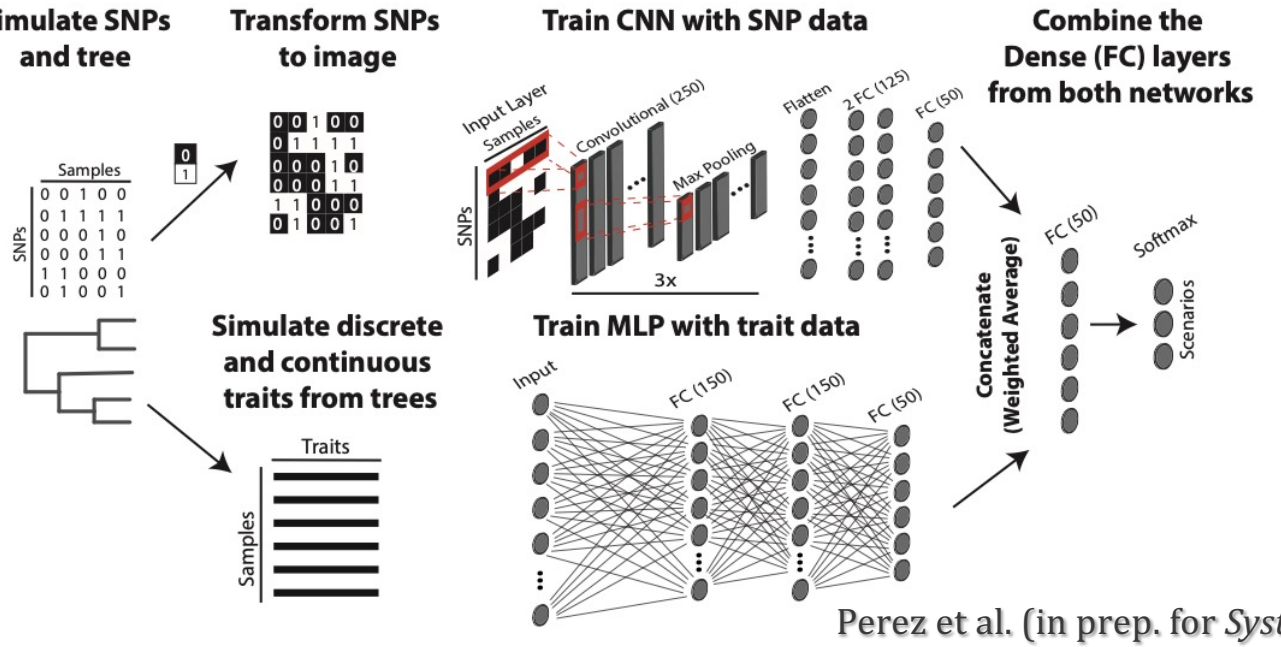
Simultaneous Inference of Past Demography and Selection from the Ancestral Recombination Graph under the Beta Coalescent

Kevin Korfmann[#,1], Thibaut Paul Patrick Sellinger[#,2,1], Fabian Freund[3,4], Matteo Fumagalli[5,6], and Aurélien Tellier[1]



Integrative Deep Learning species delimitation



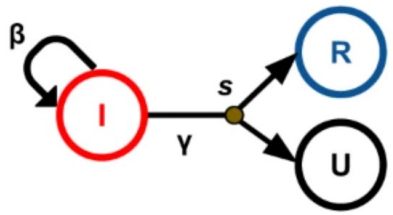


Perez et al. (in prep. for *Syst. Biol.*)

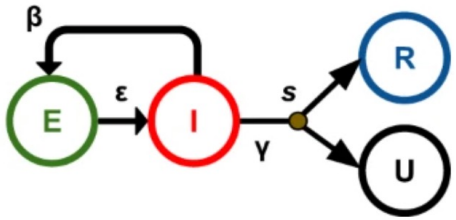
Deep Learning for **phylodynamics** and **macroevolution**

Perez & Gascuel (in prep. for *Syst. Biol.*)

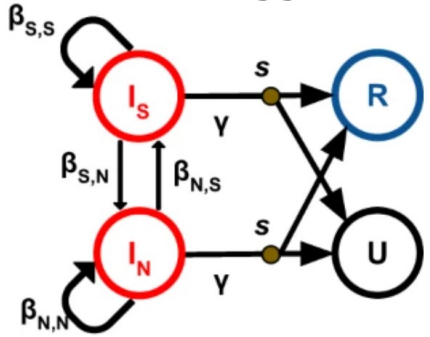
BD



BDEI

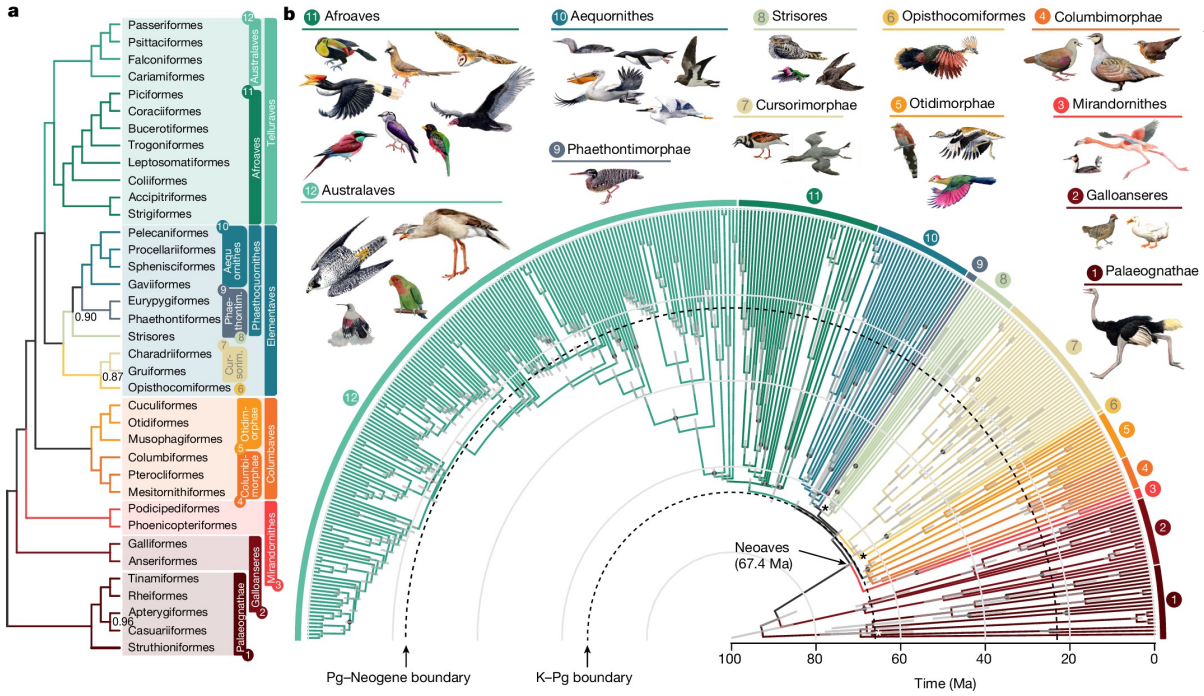


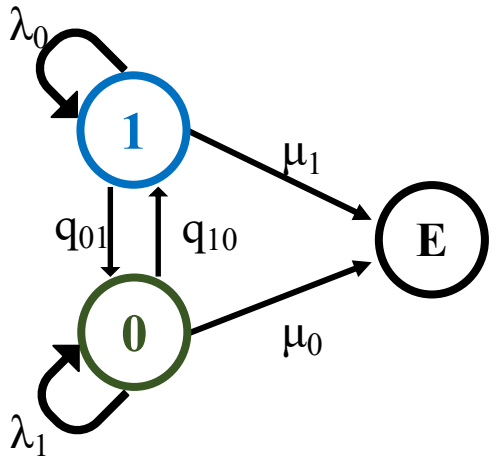
BDSS



Voznica et al. (2022) Nat Comm

Stiller et al 2024 Nature

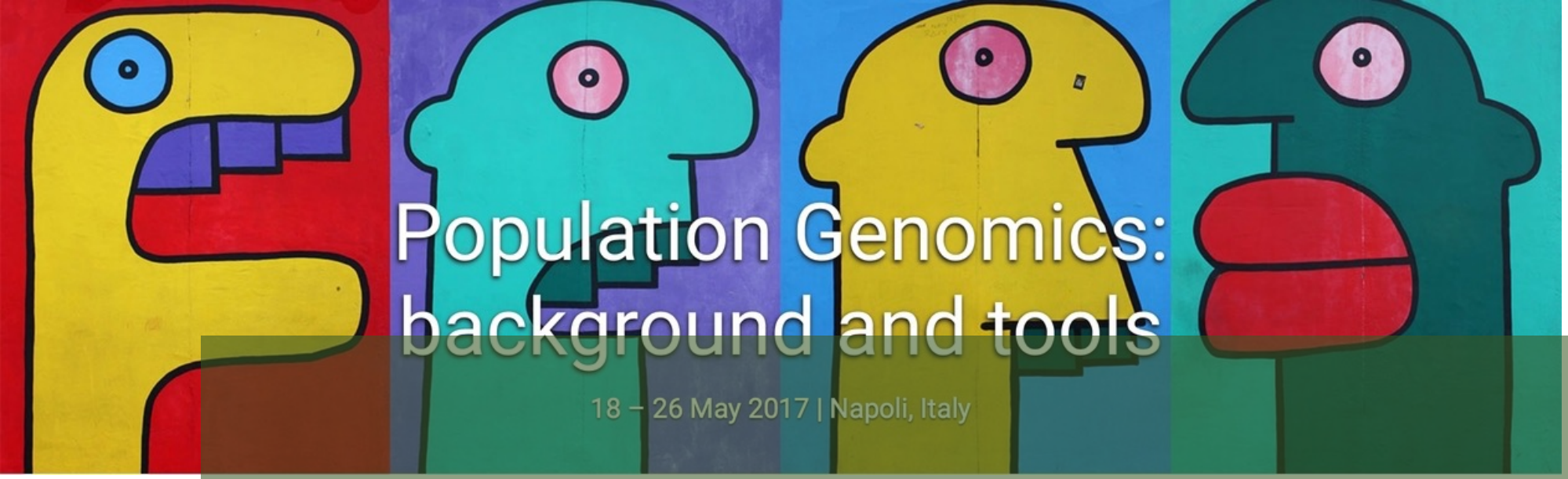




$$q = q_{01} = q_{10}$$

$$\varepsilon = \mu_0 / \lambda_0 = \mu_1 / \lambda_1$$

What's next?



Population Genomics: background and tools

18 – 26 May 2017 | Napoli, Italy

Part III: Wrapup.

Goals

- Conceive and simulate genetic data under competing demographic scenarios

- Understand deep learning background and how a CNN works

- Use CNN to detect regions with selective sweeps on real genomes

- How to use deep learning to compare demographic scenarios