Machine learning for detecting selection - practical

Matteo Fumagalli

Instructions

Just go to `https://github.com/mfumagalli/ImaGene/tree/master/Tutorials` and use `workshop.ipynb`.

We will use ImaGene `https://github.com/mfumagalli/ImaGene` which you should have already installed in your directory. If not

```
git clone https://github.com/mfumagalli/ImaGene
mamba create -n ImaGene python=3 tensorflow=2 keras=2 numpy scipy scikit-image scikit-learn matplotlib pydot pymc3 jupyterlab -y
mamba activate ImaGene
pip3 install --force --no-deps protobuf
```

You may wish to 'git pull' inside ImaGene folder to make sure you have the latest version. Then you can access its folder called Tutorials

```
cd ImaGene/Tutorials/.
```

ImaGene is simply a collection of objects and methods in python to interact with keras `keras.io/`.

ImaGene reads msms simulation files to be used for training and VCF files for deploying networks to genomic data.

ImaGene is suitable for playing with different architectures and data processing, to then use some more efficient implementations like DNADNA.

We will go over `workshop` jupyter notebook in Tutorials. Click on it and you will see this:

Detecting selection with deep learning

Examples and exercises using *keras* and *ImaGene*

This is a short tutorial to learn the basic usage of *ImaGene* which contains a series of objects in *python* to interact with *keras*.

In this practical our aim is to predict whether a given locus is under natural selection from population genomic data. We will implement a deep learning algorithm to this aim, and use *keras* for implementing the network and *ImaGene* for manipulating data. Both are accessible through *python*. In this specific example, we will perform a **binary classification** on classic example of positive selection for lactase persistence in human European populations.

Why lactase persitence? Why in Europeans?

The C/T(-13910) variant, or rs4988235, is located on chromosome 2 in the *MCM6* gene but influences the lactase *LCT* gene. This SNP is associated with the primary haplotype associated with lactose intolerance in European populations. In these populations, the common T allele is associated with lactase persistence. Individuals who are homozygous for C allele are likely to be lactose intolerant.

We extracted SNP information from a region of 80k base pairs around the target variant rs4988235 from the 1000 Genomes Project data for all unrelated individuals of CEU population (of European descent). The data is in the form of a VCF file.

In this practical, you will learn how to:

1. read data from VCF file and store it into objects,
2. run and process simulations to be used for training,
3. implement, train and evaluate the neural network,
4. deploy the trained network on your genomic data of interest.

Task

Fill in the missing values and lines in `workshop` notebook to perform binary and multiclass classification of selection.

Important notes

- You don't have to run new simulations to generate training data; data is already provided in `Data/`; you will just need to set `path_sim` variables appropriately (in practice you just need to set this variable to "./").
- If you find some of the operations to be very slow, reduce the data ("images") used; this can be done by setting `max_repl` parameter in `file_sim.read_simulations(parameter_name='selection_coeff_hetero', max_nrepl=200)`
- If the training is too slow for you, you can also reduce the number of epochs or batches of data (e.g. we created 10 batches of data, you can use half of them only).

Further questions

What happens if you retain all SNPs or remove singletons or doubletons. Is the networking learning faster with the same accuracy? Or are rare variants important for this task?

Similarly to above, what happens if you fold your data, i.e. convert ancestral/derived allelic status into major/minor? Do you obtain similar prediction accuracy?

What is the effect of modifying your architecture? You can read how to modify a sequential model in keras here `https://keras.io/guides/sequential_model/`. Note that layers are accessible with `model.layers` attribute.

Think about the difference between a binary classification, a multiclass classification and regression by reading the other tutorials. In particular, have a look at the different final layers and/or activation function.