Mushroom classification Project

Tomas Stone, *Universitat de Vic - Universitat Central de Catalunya, Spain*
Lélian Nahon, *Universitat de Vic - Universitat Central de Catalunya, Spain*
Instructor : Jordi Villa Freixa, *Universitat de Vic - Universitat Central de Catalunya, Spain*
Git-Hub Repository

Data Science and AI with Python languages and OpenAIGym

I. Introduction

A. Why ?

In this project we would like to create a guide in order to identify a mushroom, only by using facts which we could observe, such as the colour, size, shapes etc... Then our problem is the following :

Can machine learning and image analysis be used to determine the species group of mushrooms from images ?

Many guides like this already exist in the world. The new thing that we want to bring, is to find a "perfect" function which will allow anyone to recognise or eat a mushroom without taking any risk just by following our deep learning model.

Nowadays, mushroom knowledge has really evolved. We can now find technology and applications that are dedicated to identifying mushrooms. There are citizen platforms such as the iNaturalist and Mushroom Observer where everyone can upload photos, observations and critics in order to benefit the common knowledge about mushrooms. In addition to the differentiation challenges, others exist in our case. The visual similarity between species, the seasonal variation of appearances.

Moreover, our model can lead to incoherence in the process and give false results. In order to create our own database with the variables which we think are the best to analyze, we will use the following websites :

- Britannica
- Horticulture
- mushroom.world
- Wikipedia
- Kaggle

All the information we will obtain would critically help us create the database we need. Moreover the data-set of UC Irvine Machine Learning Repository, already gives us a large amount of information about mushrooms that we will use too. (data-set number : 73, name : mushroom)

B. What ? Our Hypothesis

Hypothesis 1 :

A simple method based on observing shape, location and the mushroom's colour could indicate the mushroom family species.

Hypothesis 2 :

We cannot create an efficient method. Based only on the observation of a no-expert mushroom hunter, the risks involved would be too high or in order to prevent the risk, many non-poisonous mushrooms would be discarded.

Specific objectives :

- Concentrate a large amount of data; photo from a lot of mushroom species from the 9 mushroom species families as :
  - *Agaricus*
  - *Amanita*
  - *Boletus*
  - *Cortinarius*
  - *Entoloma*
  - *Hygrocybe*
  - *Lactarius*
  - *Russula*
  - *Suillus*
- Analyze the data on python, in order to visualize the data, size, type, amount, etc...
- Organize our data in order to exploit it; resizing image, erase data which is not exploitable
- Create a model of deep learning approach with *tensorflow*
- Train this model
- Evaluating the model performance with tools like precision, recall or the f1 score
- Bonus objectives : create a user-friendly and non-expert guide to exploit our model

C. How ?

For this project we will use python, via a notebook in order to write our observations and to describe every step of our code. This solution will be findable on our github repository. To share our work the team uses Google Collab via Google Drive.

We will follow the following step:

- Creating our data collection formed of many mushroom picture, by using the previous references
- Editing the database : resizing, normalizing
- Choose our machine learning model and configure it
- Training our machine involves accuracy and have the minimum loss
- Evaluating our model, using metrics like accuracy, precision, recall and f1 score

We will use libraries such as sklearn, numpy, matplotlib, seaborn, tensorflow, os etc in order to visualize and interact with our database. We count on the amount and diversity of mushroom images we collect to train a very efficient model.

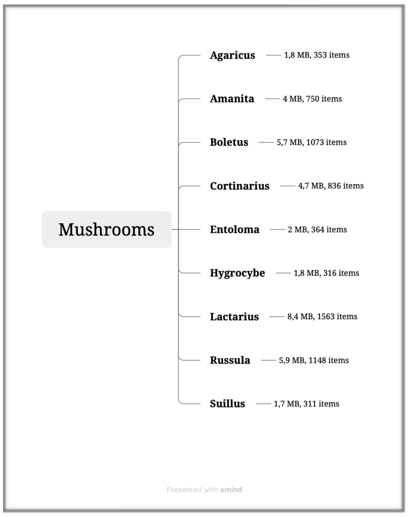
II. Dataset

Our Dataset is initially a folder with 9 folders inside for each of the 9 species earlier described. Inside these folders, we find the images of most common Northern European mushrooms. This data has been checked, provided and labeled thanks to *mycologist's society of Northern Europe*.

The origin of this data, is on *kaggle* the dataset named : *Mushrooms classification - Common genus's image.* We found it while we did our research about common knowledge and large amounts of mushroom pictures.

A. Original dataset

The folders is organized, as you can see in the following diagram :



The total size of the Mushrooms folders is 994.25 MB. All the items are in the format jpeg. That is a key point for us in order to manipulate it easily.

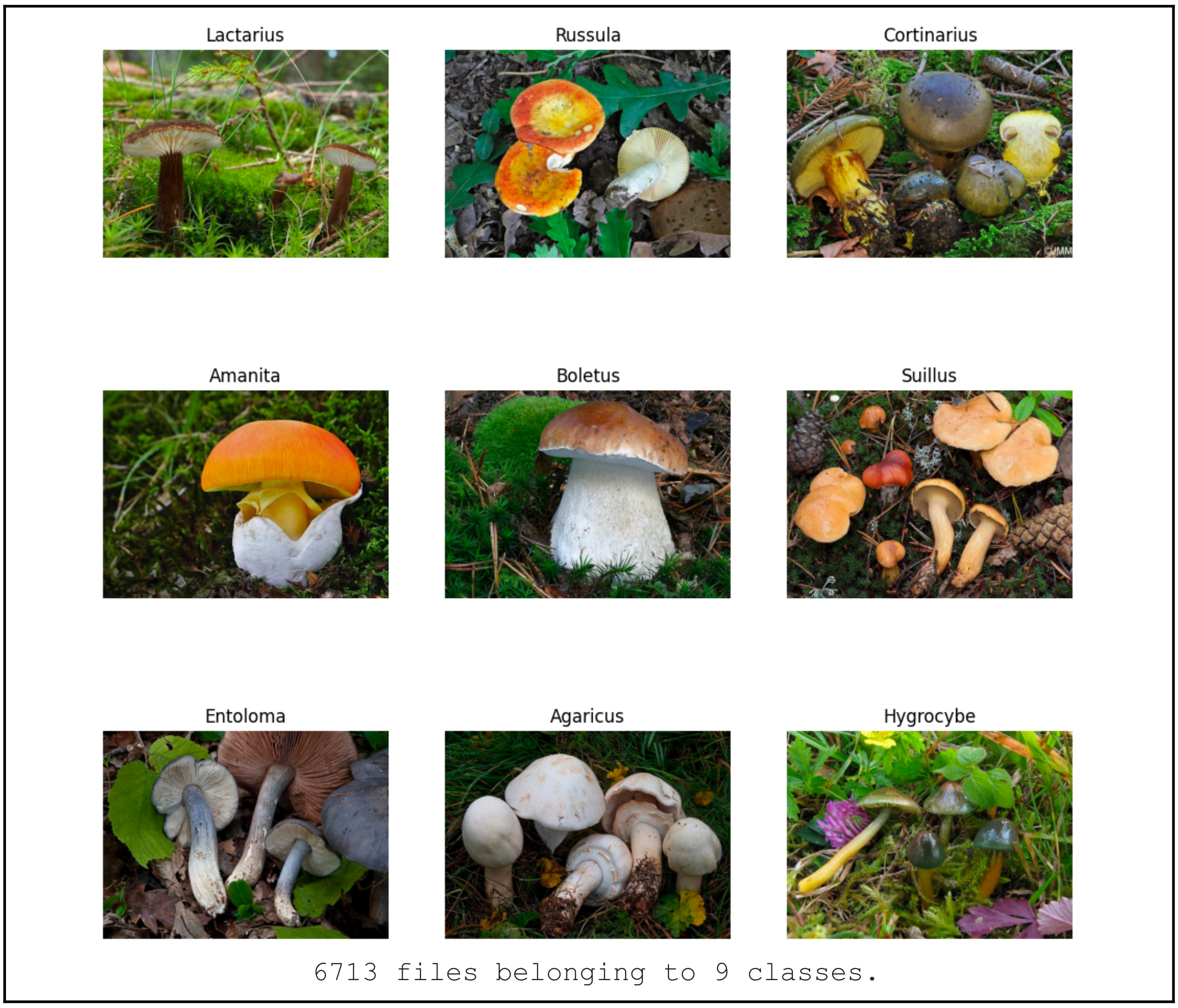
B. Dataset after editing

In order to create our deep learning model, we need that all the images have the same size. So we need to get the mean of the width and height of all the images. Like this we initially found the target size to resize all the dataset at 796x579.

Then, during the test we made, we decided to reduce by 5 this size to 159x115. Indeed, we found that the size was too big to exploit the data; our RAM wasn't strong enough and many times after almost one hour of training our program crashed. So we decided to reduce the size of our data, keep that in mind because we will speak about this choice later in our project.

The natural next step is now to resize all the data at the wanted size. Now, our dataset is now composed only of images of the same size, for a total of 6713 items. Some items could have been sorted as not usable.

Here a concrete view of sample of the data of each species in the python environment :



```
6713 files belonging to 9 classes.
```

III. Model

A. Train and Validation Datasets

To start the exploitation of the model, we need to have our dataset for the training and for the validation. To create these two dataset, we call the tensorflow's function, with parameters :

```
keras.utils.image_dataset_from_directory(
    resized,
    validation_split=0.2,
    subset="training", #then “validation”
    seed=123,
    image_size=target_size,
    batch_size=batch_size)
```

We use the data from resized, to use our brand new image, with 20% of the dataset will be used for the validation dataset. The seed is used to always have the same result, each time we perform. The image_size is the size we found earlier, a variable known as target_size. And we use a batch_size at 32 to compromise rapidity and reliability.

```
Using 5371 files for training.
…
Using 1342 files for validation.
```

B. Definition of our Convolutional Neural Network

Let's explore how we define our CNN. We use a well-known model architecture which is often from image classification.

```
model = Sequential([
    layers.Rescaling(1./255, input_shape=(159, 115, 3)),
    layers.Conv2D(16, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(32, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(64, 3, padding='same', activation='relu'),
    layers.MaxPooling2D(),
    layers.Flatten(),
    layers.Dense(10, activation='relu'),
   layers.Dense(num_classes)])
```

The yellow values are the values that we change many times. Starting with (796,579,3) to finish with (159,115,3). Obviously, we also change the target_size which is used for the training and validation data.

C. Training

The training of our model was a tough task…

As we said earlier in the document, we initially started with images that were too big. The model we have created was not efficient with the dataset we had, thus our training was too RAM memory intensive, and too long, which caused many crashes…

So we decided, after many changes in our model; like the layers.Dense passing from 128 to 10 to 9 with no special order with activation= "relu" or "softmax", adding or removing a Conv2D layer, and the different size.

The last one we use to have a result to publish is the previous one. However, the following accuracy and loss that you will see aren't at all convincing.

In the last time remaining for this work we realized that maybe our old choice of resizing the size of the image by scale of 5 is too small. Moreover maybe the layers we choose aren't the best and cannot lead to a precise result.

We hope we have time to make another model more accurate by using larger images or with news layers.

Thus, we finish with :

```
Test accuracy: 0.34500744938850403

Test loss: 2.2713656425476074
```

IV. Conclusion

A. Our model

Finally, we have a model, which is not accurate at all. But as we previously said, we want to try to have another one with a better accuracy and then the minimum of loss.

We cannot use it to classify mushroom species only with the pictures of it. At least we could do it but with an accuracy less than 50% it would not help anyone.

It is therefore logical that this model cannot be extended to identify if a mushroom is poisonous or not our initial goal.

B. Ethic

Furthermore, the result we obtained made us ask ourselves some questions.

How much confidence do we have to be confident in an application or program of this type ?

Is it ethical to create a guide which will procure a classification of species that can lead humans to disease or even death, despite warning ?

As we see with our model that mustn't be followed, is the existing guide that is worse than ours ?

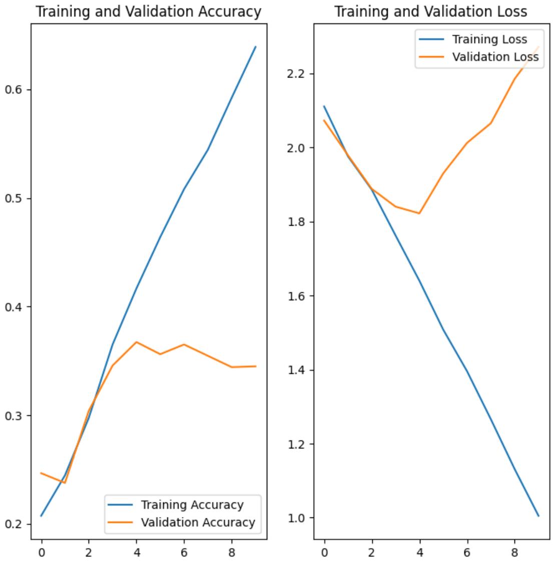
In this order we did some research and we found that there exists some organization. Like the app store which verifies applications, maybe in research of malware or virus, and not veracity of the program. Or CISSP (Certified Information Systems Security Professional), which is also used in security verification.

Thus, we thought that an organization or maybe a program which could certificate the veracity of a proposed classification or something like that could exist.

C. Our Experience

We are very happy to have experimented with the creation of a deep learning project. I think we thought it would lead more easily to a best result as it was shown of course. However, this team likes to make research in the deep common knowledge of the internet.

More precisely in this project, we would love to make another model with better accuracy and understand why the current and old model behave like they do.



As we can see here on the graph from the last model. Why the training loss and training accuracy improve when the validation behaves logically.
Maybe we are at the beginning, because we dwell on details of lesser importance and we need to explore further more.