Lab Work 3

André Pedrosa [85098], Filipe Pires [85122], João Alegria [85048]

Algorithmic Information Theory
Department of Electronics, Telecommunications and Informatics
University of Aveiro

December 29, 2019

1 Introduction

This report aims to describe the work developed for the third and final assignment of the course of 'Algorithmic Information Theory', explaining all scripts developed by us and presenting the results we considered most relevant Dataset regarding the quality of the solutions.

The programs implemented in Shell have the purpose of .....Dataset Dataset Along with the description of the solutions, we also discuss ..... All code developed is publicly accessible in our GitHub repository:Dataset `https://github.com/joao-alegria/TAI` .Dataset

2 Image Classification

The challenge we are tackling in this paper is the one presented in (*1*), which is basically a image classification problem with a special approach. Instead of doing the process chosen by most people in charge of solving similar problems, consisting in using some from of feature extraction or some kind of feature learning commonly achieved through some type of machine learning algorithm, the launched challenge aims to reach a similar goal taking only in consideration the notion of algorithmic information.

Algorithmic information, also known as algorithmic entropy or Kolmogorov complexity, is the measure of information in a given data portion based on the theoretical algorithm that would be necessary to create that same data portion. Being a quite abstract definition, an example can clarify the concept: taking in consideration the string A="0101010101" and string B="0010100110", it is much simpler to define string A than string B, since for A we can simple say that it is the string "01" repeated 5 times, where for string B it's harder and we probably need to say the entire string to describe it. This means that in reality string A as less information than string B. To achieve the algorithmic information of each one, the thought process is the same but now the string descriptions are made with algorithms, where the string can be defined be the following algorithms:

```
def stringA():
  for i in range(5):
    print("01", end="")
```

```
def stringB():
  for letter in "0010100110":
    print(letter, end="")
```

Which by a quick iteration analysis its obvious that `stringA()` is less complex than `stringB()`, needing only 5 iterations compared with 10 iterations, respectively, confirming that string B has more information than string A. This can be directly correlated with data compression, since compressors take advantage of the data repetition as dispensable data, and thereby compress the information by removing repetition but storing an indicator of that repetition.

With this concepts defined it's more clear how the image classification will be done. Using the Normalized Information Distance defined by:

$$NID(x,y) = \frac{max\{K(x|y), K(y|x)\}}{max\{K(x), K(y)\}} \tag{1}$$

Where *K(x)* defines the Kolmogorov complexity of the string x and *K(x—y)* defines the Kolmogorov complexity of the string x given string y. Unfortunately, because the Kolmogorov complexity is non-computable this formula is only theoretical since there is no algorithm that can compute any string complexity every time. For that reason compressors are used to achieve

approximations of the formula result. The studied approximations are Normalized Compression Distance(NCD) and Normalized Conditional Compression Distance(NCCD) that will be further detailed in the following subsections.

2.1 NCD Approximation

....

2.2 NCCD Approximation

....

3 Dataset

Lorem ipsum ...

4 Experiment

Lorem ipsum ...

4.1 Tested Compressors

.....

5 Results & Discussion

Lorem ipsum ...

6 Conclusions

After completing the assignment, we drew a few conclusions regarding our solutions and .....

Lorem ipsum ...

In terms of code organization and readability, we made sure our repository was as well structured as possible and our code properly commented and documented. The base folder contains a *README* file for basic instructions. All code is in the *src* folder.

References

1. Armando J. Pinho, *AIT: Lab Work no.3*, University of Aveiro, 2019/20.