Neural Grammar

Federico Pavesi

March 6, 2024

1 Introduction

For sake of clarity, we shall define two fundamental concepts which are at the base of the reasoning we are doing. When we say 'training' we refer to the process of training a neural network, which shall not be confused with 'learning', which instead is referred to the procedure of learning latent representations of neural network architectures.

A Neural Network can be seen as a complex structure representing the flow of input data through

In the context of Neural Architecture Search (NAS), one approach to deal with the search over the space of Neural Architectures is to define context-free grammar (ADD REFERENCES), such that we can define each neural architecture by its unique codification into the grammar space. In particular, if we define a network with an architecture $\omega \in \mathcal{A}$ where $\mathcal{A}$ is the class of all 'feasible' architectures, we can use the map $G : \mathcal{A} \mapsto \mathcal{G}$, with $\mathcal{G}$ is the space of codes encoded by the grammar, to pass from a neural architecture to a grammar code.

$$\nu = G(\omega) \tag{1}$$

Where ν is the grammar code associated to ω. This map is bijective, which is fundamental to allow to go back and forth from the space of architectures, in which we can train neural networks but where it is hard to learn, to the space of grammar codes where we have the opposite situation.

$$\omega = G^{-1}(\nu) \tag{2}$$

2 Neural Grammar

The grammar we implemented is not particularly complex but it has the downturn we are not able to account for all possible neural architectures. We refer the reader to

3 Appendix A: Neural Grammar

3.1 Appendix A.1: Linear Grammar

We shall here define the pseudocode for encoding linear neural networks (also known as dense networks or Multi Layer Perceptrons). Notice we define with $L \subset \mathbb{N}$ the length of the neural networks (i.e., the number of subsequent blocks) which, of course, should be finite. Also, if we want to refer to a specific block in the architecture, we write $\omega[i]$ with $i \in L$.

Algorithm 1 Encoding Linear Neural Networks

Input: MLP ω in $\mathcal{A}$
Output: Grammar code ν associated with ω

```
for i in L do
    if ω[i] is a linear module and i = 1 then        ▷ The first linear layer
    end if
end for
```