Complex Neural Arithmetic Units

Niklas Heim, Václav Šmídl, Tomáš Pevný
Artificial Intelligence Center
Czech Technical University
Prague, CZ 120 00
{niklas.heim, vasek.smidl, tomas.pevny}@aic.fel.cvut.cz

Abstract

This is the abstract.

1 Introduction

We fuse the ideas of the *Neural Multiplication Unit* (NMU) by Madsen and Johansen [2020] and the *Neural Arithmetic Logic Unit* (NALU) by Trask et al. [2018].

The NALU in its original form is capable of addition, subtraction, and multiplication of **positive**, high-dimensional vectors. However, for small inputs convergence is complicated by the use of the logarithm in the NALU. The logarithm further prevents the NALU of processing negative inputs correctly.

The NMU fixes both the previous issues, but can only multiply two inputs, and not represent arbitrary power functions. Addition is taken care of by the *Neural Addition Unit* (NAU), which is essentially a dense layer without bias. A chain of NAU and NMU can thus learn the task defined in Eq. 1.

$$y = (x_1 + x_2) \cdot (x_1 + x_2 + x_3 + x_4) \tag{1}$$

2 Complex Neural Multiplication Unit

Our *complex NMU* (NMUX) can learn arbitrary power functions while still being able to correctly deal with negative inputs. The NMUX layer is defined by

$$k_i = \begin{cases} 0 & x_i \leq 0 \\ \pi & x_i > 0 \end{cases} \tag{2}$$

$$\boldsymbol{r} = |\boldsymbol{x}| \tag{3}$$

$$\boldsymbol{m} = \exp(\boldsymbol{M} \log(\boldsymbol{r})) \odot \cos(\boldsymbol{M}\boldsymbol{k}) \tag{4}$$

where $\boldsymbol{k}$ is a vector that is zero where $\boldsymbol{x}$ is positive and π where it is negative, $\boldsymbol{r}$ is the elementwise absolute value, and $\boldsymbol{M}$ the multiplication matrix that we want to learn.

The NMUX is inspired by the multiplication part of the NALU, extended to negative inputs by leveraging taking advantage of the complex logarithm. The multiplication part of the NALU reads:

$$\boldsymbol{m} = \exp(\boldsymbol{M} \log(|\boldsymbol{x}| + \epsilon)), \tag{5}$$

where ϵ is a small constant that ensures positive inputs to the real logarithm. If use the complex logarithm, this constant becomes superfluous, but we end up with a complex output in case of negative x_i. In general, for any complex number z[1]

$$\log(z) = \log\left(r \cdot e^{i\varphi}\right) = \log(r) + i\varphi. \tag{6}$$

[1]Note that complex numbers can be represented in the polar, complex plane where $z = re^{i\varphi}$ with $r > 0$ and $\varphi \in (0, 2\pi)$.

Preprint. Under review.

However, as z is assumed to be real, we can simplify as follows:

$$\log(z) = \log(r) + ik\pi, \tag{7}$$

where $k \in [0, 1]$.

$$\begin{aligned} \boldsymbol{m} &= \exp(\boldsymbol{M} \log(\boldsymbol{x})) && (8)\\ &= \exp(\boldsymbol{M}(\log(\boldsymbol{r}) + i\pi\boldsymbol{k})) && (9) \end{aligned}$$

$$\begin{aligned} \boldsymbol{m}_{\text{re}} &= \text{real}(\exp(\boldsymbol{M} \log \boldsymbol{r} + i\pi\boldsymbol{M}\boldsymbol{k})) && (10)\\ &= \text{real}(\exp(\boldsymbol{M} \log \boldsymbol{r}) \odot (\cos(\pi\boldsymbol{M}\boldsymbol{k}) + i \sin(\pi\boldsymbol{M}\boldsymbol{k}))) && (11)\\ &= \exp(\boldsymbol{M} \log \boldsymbol{r}) \odot \cos(\pi\boldsymbol{M}\boldsymbol{k}) && (12) \end{aligned}$$

3 Experimetns

3.1 10 param func

3.2 Gradient surfaces

References

Andreas Madsen and Alexander Rosenberg Johansen. Neural Arithmetic Units. page 31, 2020.

Andrew Trask, Felix Hill, Scott Reed, Jack Rae, Chris Dyer, and Phil Blunsom. Neural Arithmetic Logic Units. *arXiv:1808.00508 [cs]*, August 2018. URL `http://arxiv.org/abs/1808.00508`. arXiv: 1808.00508.