FACULTY OF SCIENCE
UNIVERSITY OF COPENHAGEN

DEPARTMENT OF COMPUTER SCIENCE

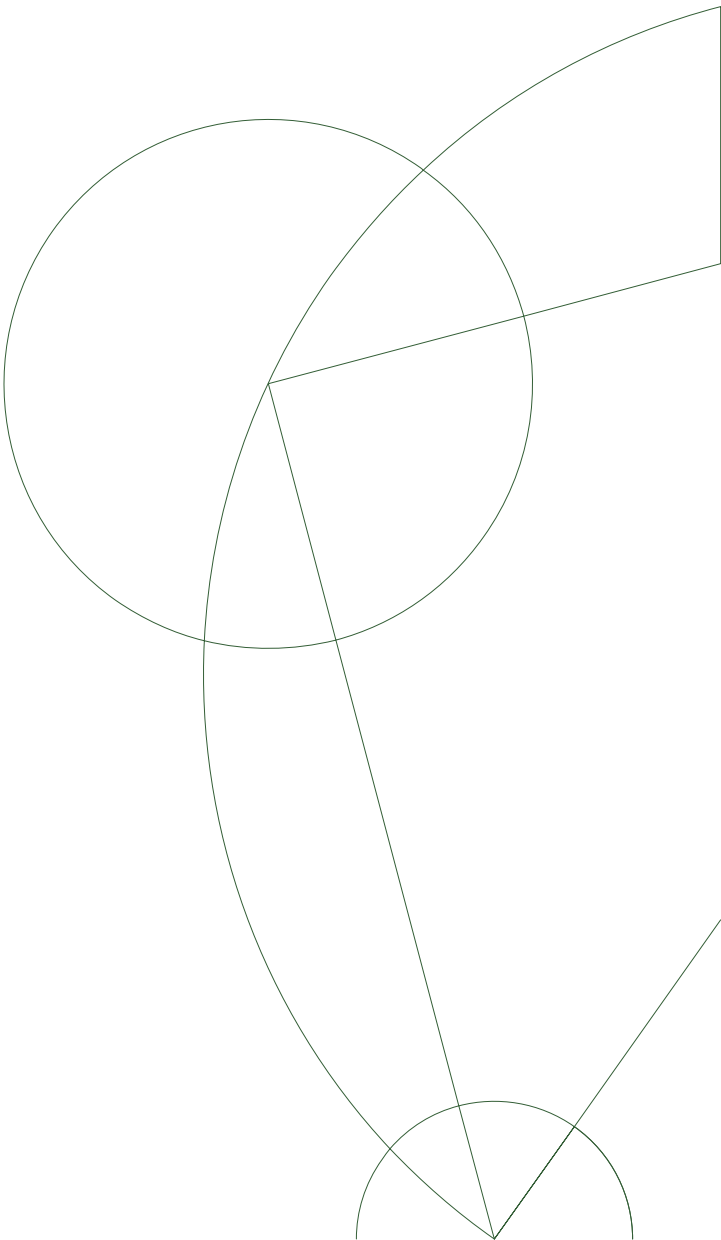
Modelling learning systems

A DSL for cognitive neuroscientists

Jens Egholm Pedersen <xtp778@alumni.ku.dk>

Supervisor

Martin Elsman <mael@di.ku.dk>



1 Introduction

In the past years machine learning has surpassed humans in some recognition tasks, and the development shows no signs of slowing down. These developments are however based on relatively old research on neural networks (Nilsson 2009; Russell and Norvig 2002). Newer investigation into rehabilitation and learning indicates that such networks alone cannot account for the same amount of learning that happens in the brain (Mogensen 2011; Block 2007; Russell and Norvig 2002; Moravec 1998; Dennett 2017). For that reason the breakthroughs in machine learning are hard to transfer to the domain of cognitive neuroscience.

As an attempt to remedy this, this project sets out to define a domain-specific language (DSL) that is capable of representing the concepts of learning systems within the domain of neuroscience. The latter part of the paper validates this DSL through the modelling of a small learning task. The benchmark will be written in Futhark and compiled to the OpenCL standard, but the DSL abstraction allows it to be executed on any machine architecture.

The goal is for the DSL to lay the foundation for a more accurate scientific representation of learning and learning concepts, serving as a more approachable simulation tool for cognitive neuroscience.

1.1 Problem statement

Building on theories and concepts of the domain of cognitive neuroscience this paper examines the hypothesis that *the DSL presented in this paper can model meaningful machine learning tasks for the cognitive neurosciences, agnostic of the learning system implementation*. The paper will approach this in two steps:

1. Defining and implementing a DSL abstraction for the expression of learning tasks, based on the REF model from (Mogensen 2011).

2. Testing the DSL by expressing a learning task in a Krechevsky T-maze (Krechevsky 1932), backed by a traditional machine learning implementation in Futhark.

2 Theory

This section accounts for the theoretical foundation of paper and is divided into three parts. The first part concerns the broad topic of computation and learning in neural systems as seen from the perspective of computational neuroscience. By focusing on cognition, plasticity, learning and rehabilitation, it derives the necessary and sufficient language abstractions to capture the complexity of the domain. The second part introduces traditional machine learning from the perspective of computer science. These concepts will be applied in the validation phase of learning model abstractions in section 4. In the final part the theoretical background for linguistic abstractions and the construction of domain specific languages will be treated.

2.1 Computation and learning in neural systems

> Activity-dependent synaptic plasticity is widely believed to be the basic phenomenon underlying learning and memory (Dayan and Abbot 2001).

Commonly referred to as *what fires together, wires together*, Hebbian learning suggests that synaptic connections from neuron A to neuron B are strengthened or weakened when neuron A excites or inhibits the chance of firing neuron B respectively (Dayan and Abbot 2001). Hebbian learning is believed to play a large part in the plastic nature of the brain, especially within learning and memory formation (Dayan and Abbot 2001; Johnston 2009; Robertson and Murre 1999).

Reorganisation of elementary functions

(Robertson and Murre 1999) studied patients during rehabilitation of brain damage and conjectured that learning — whether when the brain acquires new information or recovers from lost information — occurs based on the structural changes induced by the Hebbian principle (Robertson and Murre 1999). They further concluded that the damaged brain areas regenerate themselves based on this principle (Robertson and Murre 1999). (Mogensen 2011) refutes this point by claiming that, while there may be some synaptogenesis,

the rehabilitation mainly occurs when other parts of the brain learns to take over the lost functions (Mogensen 2011).

(Mogensen 2011) arrives at a theoretical framework which divides the brain into localized and highly specialised, basic information processing elementary functions (EF). These modular functions are contained in a *substructure* or *local circuit* within the brain (Mogensen 2011).

Multiple EFs interact to form algorithmic strategies (AS), established as a consequence of learning and experiencing (Mogensen 2011). An algorithmic strategy combines the capacity of multiple EFs into a single - and for the AS appropriate - response (Mogensen 2011; Mogensen 2012).

The EF and AS interplay to create what Mogensen dubbed '*surface phenomena*', which manifests the behaviour of the system (Mogensen 2011). Surface phenomena are the product of applying an AS to a particular problem, and Mogensen hypothesised that a given AS is evaluated for every success or failure of the surface behaviour predicted by that AS (Mogensen 2011). Such evaluation either strengthens the AS's association with the given behaviour scenario, or weakens et, potentially starting a search for another AS to perform the task instead (Mogensen 2011). According to Mogensen these changes are controlled by a specialised AS dubbed the *Supervisory Attentional System* (SAS) from (Norman and Shallice 1986), manifested through neuroplastic changes in the synaptic connections within the SAS (Mogensen 2011).

Behaviour in the REF model is thus defined as the response of a single AS (that, in turn, consists of a number of EFs) to a given task (Mogensen 2011; Mogensen 2012).

An elaboration to the REF model arrived in the form of the REFGEN model (general reorganisation of elementary functions) (Mogensen and Overgaard 2017). The REFGEN model further explains the feedback mechanisms of the SAS to account for the 'learning' or adaptive feature of neural systems, by introducing two new concepts: the goal algorithmic strategy (GAS) and the comparator (Mogensen and Overgaard 2017; Mogensen 2012).

In this new framework the SAS is tasked with maintaining the current state of the system, while the GAS reflects the goal towards which it is desired to to move (for instance the exit condition in a maze) (Mogensen and Overgaard

2017). For this to be useful a comparator is needed to constantly compare the SAS and GAS (the current state versus the goal), such as to select the optimal AS for the task at hand (Mogensen and Overgaard 2017). The feedback (back-propagation) from the actuation on the surrounding world is received by the comparator, who will assert influence on the SAS and GAS to better account for the new reality (Mogensen and Overgaard 2017).

2.2 Reinforcement learning

2.3 Language abstractions

3 Volr: A DSL for learning systems

4 Case study: applying Volr in a Krechevsky maze

Glossary

Futhark A programming language geared towards performance in parallel environment such as graphics processors (GPUs). Futhark is a purely functional array language and is developed by HIPERFIT research center under the Department of Computer Science at the University of Copenhagen (DIKU).. 2

OpenCL An open standard for cross-platform parallel programming, which allows software to be executed on CPUs, GPUs or other processors or hardware accelerators. See `https://www.khronos.org/opencl/`. 2

Bibliography

Block, Ned (2007). "Consciousness, accessibility, and the mesh between psychology and neuroscience". In: *Behavioral and Brain Sciences* 30.5-6, pp. 481–499. DOI: 10.1017/S0140525X07002786.

Dayan, Peter and L. F. Abbot (2001). *Theoretical neuroscience - Computational and Mathematical Modeling of Neural Systems*. MIT Press. ISBN: 978-0-262-04199-7.

Dennett, Daniel C. (2017). *From bacteria to Bach and back*. ISBN: 978-0-393-24207-2.

Johnston, Michael V. (2009). "Plasticity in the developing brain: Implications for rehabilitation". In: *Developmental Disabilities Research Reviews* 15.2, pp. 94–101. ISSN: 1940-5529. DOI: 10.1002/ddrr.64. URL: http://dx.doi.org/10.1002/ddrr.64.

Krechevsky, I (1932). "Hypotheses in Rats". In: 39, pp. 516–532.

Mogensen, J. (2011). "Reorganization of the Injured Brain: Implications for Studies of the Neural Substrate of Cognition". In: *Frontiers in Psychology* 2, p. 7. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2011.00007. URL: https://www.frontiersin.org/article/10.3389/fpsyg.2011.00007.

— (2012). "Reorganization of Elementary Functions (REF) after brain injury: implications for the therapeutic interventions and prognosis of brain in-

jured patients suffering cognitive impairments". In: *Brain Damage: Causes, Management and Prognosis* 1. Ed. by A. J. Schäfer and J. Müller, pp. 1–40.

Mogensen, J. and M. Overgaard (2017). "Reorganization of the Connectivity between Elementary Functions – A Model Relating Conscious States to Neural Connections". In: *Frontiers in Psychology* 8, p. 625. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2017.00625. URL: https://www.frontiersin.org/article/10.3389/fpsyg.2017.00625.

Moravec, Hans (1998). "When will computer hardware match the human brain". In: *Journal of Transhumanism* 1.

Nilsson, Nils J. (2009). *The Quest for Artificial Intelligence*. 1st. New York, NY, USA: Cambridge University Press. ISBN: 9780521122931.

Norman, Donald and Tim Shallice (1986). *Attention to Action*.

Robertson, Ian H. and Jaap M. J. Murre (1999). "Rehabilitation of Brain Damage: Brain Plasticity and Principles of Guided Recovery". In: *Psychological Bulletin* 125.5, pp. 544–575.

Russell, Stuart J. and Peter Norvig (2002). *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall. ISBN: 0137903952. URL: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20%5C&path=ASIN/0137903952.

5 Appendix A: Volr EBNF

```
model = ( stimulus | strategy | response )
      , { [ space ] , "," , [ space ] , ( stimulus | strategy | response ) };

stimulus = "stimulus" , [ space ] , name , [ space ] , dimensionality;
response = "response" , [ space ] , dimensionality , [ space ] , input
         , [ space ] , { "select" , [ space ] , ( "random" | "best" ) };
strategy = "strategy" , [ space ] , name , [ space ] , input
         , [ space ] , ( functions | function , [ space ] , { function } );

functions = "functions" , [ space ] , dimensionality;
function = "function" , [ space ] , name , dimensionality;
```

```
  input = "from" , [ space ] , name , { [ space ] , name };
  dimensionality = "[" , [ space ] , integer , [ space ] , "]";

  name list = name , { [ space ] , "," , name };
  name = letter , { letter | digit };
  letter = ? non-whitespace Unicode character ?;
  space = space character , { space character };
  space character = ? white space character ?;

  number = integer , ["." , digit , {digit}];
  integer = ["-"] , digit , {digit};
  digit = "0" | "1" | "2" | "3" | "4"
        | "5" | "6" | "7" | "8" | "9";
```

6 Appendix B: Learning network with two features in Volr

```
stimulus input [2]

strategy s1 from input
  functions [10]

strategy s2 from s1
  functions [10]

strategy s3 from s1
  functions [5]

response [1] from s2, s3
  select best
```