

# HYPERDIMENSIONAL COMPUTING FOR PROTEIN LANGUAGE MODELING

...

Michael Fatjanov

Student ID: ...

Supervisor(s): Prof. Dr. Bernard De Baets

A dissertation submitted to Ghent University in partial fulfilment of the requirements for the degree of master in Bioinformatics.

Academic year: 2022-2023

De auteur en promotor geven de toelating deze scriptie voor consultatie beschikbaar te stellen en delen ervan te kopiëren voor persoonlijk gebruik. Elk ander gebruik valt onder de beperkingen van het auteursrecht, in het bijzonder met betrekking tot de verplichting uitdrukkelijk de bron te vermelden bij het aanhalen van resultaten uit deze scriptie.

The author and promoter give the permission to use this thesis for consultation and to copy parts of it for personal use. Every other use is subject to the copyright laws, more specifically the source must be extensively specified when using results from this thesis.

Gent, FILL IN THE DATE

The promotor,

The author,

Prof. Dr. Bernard De Baets

Michael Fatjanov

# **ACKNOWLEDGEMENTS**

Thank you, all of you!



# **CONTENTS**

<b>Acknowledgements</b>	<b>i</b>
<b>Contents</b>	<b>iii</b>
<b>Nederlandse samenvatting</b>	<b>v</b>
<b>Summary</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
<b>Bibliography</b>	<b>4</b>



# **SAMENVATTING**

nederlandse samenvatting





# **SUMMARY**

insert english summary here...



# **1. INTRODUCTION**

**Digital biology, protein sequence research and traditional bioinformatics tools**

**State-of-the-art, deep learning and protein language modeling**

**Hyperdimensional computing**

Hyperdimensional computing is a relatively new paradigm of computing developed by Kanerva. Kanerva (2009) that tries to mimick the workings of a (human) brain by computing with vectors of tens of thousands elements long, so in the realm of hyperdimensionality. The human brain consists of about 100 billion neurons (nerve cells) and 1000 trillion synapses that connect these neurons. Each neuron is connected up to 10000 other neurons, creating massive circuits. This is likely fundamental to the workings of the human brain and what separates our brains from modern von Neumann computer architectures which operate on 8 to 64 bit vectors. This becomes clear when we compare the relative simplicity for a human to learn a language compared to computers which require a large and complicated set of arithmetic operations in the form of deep learning networks together with terabytes of data and thousands of Watts of compute power to try to come close mastering a language whilst a human can recognize other languages relatively easy when they don't even speak it. Likewise languages, we can very easily memorize and compare other intrinsically complex and contextual concepts such as images. A computer would have a hard time finding similarity between a set of images and faces because this requires very complex machine learning models. The human brain can do this all with a huge efficiency by consuming only roughly 20 W of energy.

---

To achieve this kind of flexible intelligence, we might have to step away from the restrictive von Neumann architecture and so Kanerva proposes the use of hyperdimensional vectors, a different form of representation for entities by which a computer can compute with. The use of models based on high dimensionality is not entirely new and is being explored since the 1990s. Some of these earlier models include Holographic Reduced Representations Plate (1995), Spatter Code Kanerva (1994) etc. An HDV can represent anything from a scalar number to any kind of concept. This vector is initially made up of totally random elements, but with a simple set of operations which will be explained later, we can use other vectors to combine some concepts into new similar or dissimilar concepts. For example to show the essence of HDC and how it tries to simulate the brain, we can compare the concept of a *table* to the concept of a *broccoli*. We would not immediately conclude that they are in any way similar but as humans we can trace back *table* to *plate* which has some similarities with *food* from which we can easily extract the concept of *broccoli*. These kind of operations are not very obvious for a classical computer but easy for us humans.

The elements in an HDV can be made up from binary bits like in classical computing but also of bipolar or real numbers. The choice of the nature of the elements has also implications on the nature of the different operations and on the results. Highly efficient bit operations could be used on binary vectors but then the amount of information stored in such a vector would be drastically lessened compared to bipolar or real vectors, leading to a lower accuracy.

An initial HDV is made up totally random. This *holistic* or *holographic* representation of a concept smeared out over a vector consisting of thousands of bits gives rise to interesting properties such as its robustness. These kind of systems are very tolerant to noise and failure of bits since we introduce a lot of redundancy in the vector just by stochastics. This is very unlike classical computing where every bit counts and one failure in a bit can lead to disasters.

## **Operations on hyperdimensional vectors**

The interesting properties of HDC are based on only four basic operations we can perform on HDVs. We will discuss these for bipolar and binary vectors.

### Addition

Also referred to as *bundling*, the element-wise addition as in equation 1.1 of  $n$  input vectors  $\{X_1 + X_2 + \dots + X_n\}$  creates a vector  $X$  that is maximally similar to the input vectors.

$$X = X_1 + X_2 + \dots + X_n \quad (1.1)$$

For bipolar vectors this is straightforward. The input vectors are added element-wise but the resulting vector is restricted to a bipolar nature too, thus to contain only  $-1$ ,  $1$  and allowing  $0$  for elements that are in disagreement as shown in the following 6-dimensional example.

$X_1 =$	$+1$	$-1$	$+1$	$+1$	$-1$	$-1$
$X_2 =$	$+1$	$+1$	$+1$	$-1$	$-1$	$-1$
$X_3 =$	$-1$	$-1$	$+1$	$+1$	$-1$	$+1$
$X_4 =$	$-1$	$-1$	$-1$	$+1$	$-1$	$+1$
<hr/>						
$X_1 + X_2 + X_3 + X_4 =$	$0$	$-1$	$+1$	$+1$	$-1$	$0$

For binary vectors, the vectors are element-wise bundled based on the majority element. This is no problem if an odd number of input vectors are considered but ambiguity rises when bundling an even set of vectors. This can be solved by setting the element in question randomly. Schmuck et al. (2019) Another possibility is to add another random vector however this may seem to add the most noise, especially when bundling a low number of vectors.

### Multiplication

Also referred to as *binding*, we can element-wise multiply two vectors resulting in a vector maximally dissimilar to the input vectors. Vectors  $X$  and  $Y$  are bound together forming  $Z = X * Y$  with  $Z$  being orthogonal to  $X$  and  $Y$ . This *binding* operation translates to a simple arithmetic element-wise multiplication for bipolar vectors. For binary vectors this represents a *XOR* bit-operation shown as follows.



# **BIBLIOGRAPHY**

Kanerva, P. (1994). The spatter code for encoding concepts at many levels. pages 226–229.

Kanerva, P. (2009). Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors. *Cognitive Computation*, 1(2):139–159.

Plate, T. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6(3):623–641.

Schmuck, M., Benini, L., and Rahimi, A. (2019). Hardware optimizations of dense binary hyperdimensional computing: Rematerialization of hyper-vectors, binarized bundling, and combinational associative memory. *J. Emerg. Technol. Comput. Syst.*, 15(4).