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

SAMENVATTING

nederlandse samenvatting

SUMMARY

insert english summary here...

1. [INTRODUCTION](#)

1.1 Big picture and traditional bioinformatics tools for protein research

Proteins are an essential part of molecular biology and are responsible for a wide variety of functions. Far too wide to discuss here because they are one of the building blocks that make up life, hence a lot of effort has gone towards trying to understand the functions of protein and disruptions in its mechanisms that lead to many kinds of diseases. In spite of that, for a large fraction of the approximately 20000 human proteins, the structures and functions remain still unknown. To start, proteins are composed of a linear chain of amino acids (AA) with a length ranging from 50 to tens of thousands of AAs, all connected by peptide bonds into a polypeptide. This is also referred to as the *primary structure* of a protein.(1) A sequence of amino acids is mostly determined by the genetic code without considering post-translational and post-transcriptional modifications etc. In the genetic code of all living organisms, there are 20 different kinds of amino acids coded which make up the 'language' of proteins. The current state-of-the-art methods for the identification of protein sequences are *de novo sequencing* algorithms applied to tandem mass spectrometry data.(2)

It is intuitive to represent a protein as a sequence of letters with each letter corresponding to an amino acid. Likewise to natural languages, we can find common elements between naturally evolved proteins. These motifs and domains are essential to many biological processes and can easily be represented as words, phrases and sentences of amino acids. A protein also consists of much more than a mere sequence, however. It is a 3-dimensional structure with a predetermined form and function. While a protein's structure and function are dynamic and dependent on its surroundings such as the cellular state and other proteins and molecules akin to natural languages, it is still defined by its underlying sequence. This means that a lot of the 3D-structural and functional information of a protein should be retriev-

able from its amino acid sequence.(3) The most common way to determine the 3D structure of a protein has remained to be X-ray crystallography for more than half a century (4), with cyro-electron microscopy now catching up rapidly.(5) However, these kinds of laboratory approaches for structure determination of proteins are not simple, expensive and in some cases not possible for the protein in question. On top of that, structure prediction methods are highly in demand for researchers in applications such as drug design. Therefore, a lot of effort has gone into computational methods for structure and function predictions from protein sequences.

A traditional task in bioinformatics and more specifically in the realm of protein sequence analyses is the quantification of the similarity between strings of sequences. Classical algorithms for this include the algorithm of **Needleman & Wunsch** (6) for global alignments and that of **Smith & Waterman** (7) for local alignments. These algorithms are sufficient for comparing two sequences but unfeasible for searching databases for homologous sequences, hence faster algorithms like FASTA (8) and, as of yet widely used for simple searches for similar sequences, BLAST (9) were made. To cope with the number of recorded protein sequences rising exponentially, far more compute efficient methods had to be developed like PSI-BLAST (10), HHblits (11) and MMseqs (12).

1.2   State-of-the-art protein language modeling

1.3   Hyperdimensional computing

Hyperdimensional computing (HDC) is a relatively new paradigm of computing developed by **Kanerva** (13) that tries to mimic the workings of a (human) brain by computing with vectors of tens of thousands of 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 to 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 lan-

2

guage compared to computers. Computers use a large and complicated set of arithmetic operations in the form of deep learning networks which require terabytes of data and thousands of Watts of computing power to come close to mastering a language whilst a human can recognize other languages relatively easily 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 similarities 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.

Achieving these kinds of flexible brain-like 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 (14), Spatter Code (15) etc. A hyperdimensional vector (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 *brocolli*. 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 *brocolli*. These kinds of operations are not very obvious for a classical computer but creating these semantic pathways are rather easy for humans.

The elements in an HDV can be made up of 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 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 lower accuracy.

An initial HDV is made up fully randomly. 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 kinds 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 clas-

sical computing where every bit counts and one failure in a bit can lead to disasters.

1.3.1 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.

1.3.1.1 Similarity measurement

For many kinds of problems, it will be necessary to quantify the similarity between two HDVs. The method depends on the nature of the vectors. For binary vectors, the *Hamming distance* defined as in equation 1.1 is widely used.

$$Ham(A, B) = \frac{1}{d} \sum_{i=1}^{d} 1_{A_{(i)} \neq B_{(i)}} \tag{1.1}$$

The *cosine distance* as defined in equation 1.2 is most commonly used for bipolar vectors.

$$cos(A, B) = \frac{A \cdot B}{||A|| * ||B||} \tag{1.2}$$

The results of both of these measurements are summarized in table 1.1. It

Measurement	Dissimilar	Orthogonal	Similar
Hamming distance	1	0.5	0
Cosine similarity	-1	0	1

Table 1.1: Overview of similarity measurements in HDC depending on the nature of the HDVs

is important to note that two random HDVs will be orthogonal to each other just by stochastics. Also notice that the first quantifies a distance and the latter a similarity.

1.3.1.2 Addition

Also referred to as *bundling*, the element-wise addition as in equation 1.3 of n input vectors $\{X_1 + X_2 + \cdots + X_n\}$ creates a vector X that is maximally similar to the input vectors.

$$X = X_1 + X_2 + \cdots + X_n \tag{1.3}$$

For bipolar vectors this is straightforward. The input vectors are added element-wise but the resulting vector is restricted to a bipolar nature too depending on the sign of each element, thus containing only $-1, 1$ but 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	$+1$
$X_1 + X_2 + X_3 + X_4 =$	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. (16) Another possibility is to add another random vector however this may seem to add more unnecessary noise, especially when bundling a low number of vectors. We can also reverse this by an *inverse addition*. For bipolar vectors, this means just multiplying the vector of interest by -1. A binary vector can be flipped bit-wise.

Similar to an ordinary arithmetic summation, the bundling addition of hyper-dimensional vectors is commutative so the result is not dependent on the order of addition.

$$X_1 + X_2 = X = X_2 + X_1 \tag{1.4}$$

1.3.1.3 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 being orthogonal to X and Y as shown in equation 1.5.

$$Z = X * Y \tag{1.5}$$

This *binding* operation translates to a simple arithmetic element-wise multiplication for bipolar vectors. For binary vectors, this is represented by a *XOR*

bit-operation shown as follows.

$$
\begin{array}{rcccccc}
X = & 1 & 0 & 1 & 1 & 0 & 0 \\
Y = & 1 & 1 & 0 & 1 & 0 & 1 \\
\hline
X * Y = & 0 & 1 & 1 & 0 & 0 & 1 & 0
\end{array}
$$

This operation can also be undone by multiplying with the same vector again. It is its own inverse so that

$$A * A = O \text{ where } O \text{ is a vector containing only 0s} \qquad (1.6)$$

Likewise an ordinary multiplication, this operation is commutative and distributive over additions, meaning that transforming a bundle of concepts with binding is equivalent to binding every element before bundling.

$$A = Z * (X + Y) = XZ + YZ \qquad (1.7)$$

1.3.1.4 Permutation

The permutation operation of an HDV, also known as *shifting*, is a simple reordering of the HDV. This can be random but a circular shift is widely employed (17) and makes the operation easily reversible. This results in a vector technically dissimilar from the input vector but still encoding its information. This will become important later when it will be used to encode sequential information such as tokens in a text. This operation will be denoted by Π.

$$
\begin{array}{rcccccc}
X = & 1 & 0 & 1 & 1 & 0 & 0 \\
\hline
\Pi(X) = & 0 & 1 & 0 & 1 & 1 & 0
\end{array}
$$

1.3.2 Examples

There are many interesting possibilities given the relative simplicity of all these operations. We shall illustrate some applications and examples. In the following example, the robustness of these hyperdimensional vectors is shown. Assume A, B, C, X, Y, Z to be random 10000-dimensional bipolar

hypervectors and $D = X * A + Y * B + Z * C$. We will try to retrieve A from D.

$$A' = X * D \tag{1.8}$$

$$= X * (X * A + Y * B + Z * C) \tag{1.9}$$

$$= \underbrace{X * X * A}_{A} + \underbrace{X * Y * B + X * Z * C}_{noise} \tag{1.10}$$

$$\approx A \tag{1.11}$$

This example was implemented in a Julia script, the results are illustrated in figure 1.1.
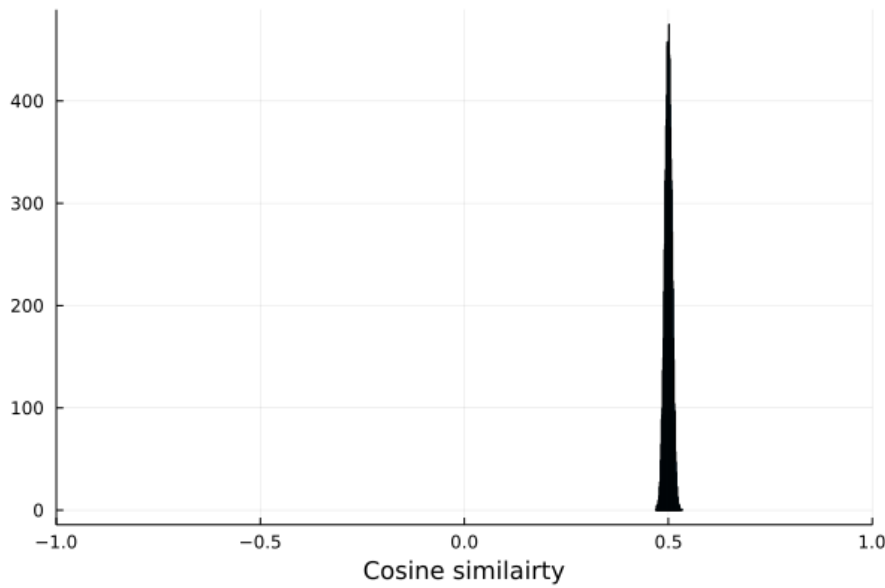

Figure 1.1: 10000 cases of random 10000-dimensional bipolar vectors are made and each time implemented following example 1.8. The resulting cosine similarities between A and A' are then plotted in a histogram.

We see that we can retrieve a lot of information with most of the cosine similarities centering around 0.5. Notice that two completely random HDVs would have a cosine similarity close to 0 just by stochastics. This result is not comparable to state-of-the-art accuracies but very efficient nonetheless as all of these calculations were done in less than 2 seconds. This same experiment was done with random binary 10000-dimensional vectors and it finished even faster as expected but retained the accuracy (similar *Hamming distance* peak around 0.25).

BIBLIOGRAPHY

[1] JOSEPH S. FRUTON. Early theories of protein structure. *Annals of the New York Academy of Sciences*, 325(1):1–20, 1979.

[2] Simone König, Wolfgang M. J. Obermann, and Johannes A. Eble. The current state-of-the-art identification of unknown proteins using mass spectrometry exemplified on de novo sequencing of a venom protease from bothrops moojeni. *Molecules*, 27(15), 2022.

[3] O.B. Ptitsyn. How does protein synthesis give rise to the 3d-structure? *FEBS Letters*, 285(2):176–181, 1991.

[4] J. C. Kendrew, G. Bodo, H. M. Dintzis, R. G. Parrish, H. Wyckoff, and D. C. Phillips. A three-dimensional model of the myoglobin molecule obtained by x-ray analysis. *Nature*, 181(4610):662–666, March 1958.

[5] Ka Man Yip, Niels Fischer, Elham Paknia, Ashwin Chari, and Holger Stark. Atomic-resolution protein structure determination by cryo-EM. *Nature*, 587(7832):157–161, October 2020.

[6] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

[7] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.

[8] David J. Lipman and William R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227(4693):1435–1441, 1985.

[9] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

[10] S. Altschul. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, September 1997.

[11] Martin Steinegger, Markus Meier, Milot Mirdita, Harald Vöhringer, Stephan J. Haunsberger, and Johannes Söding. HH-suite3 for fast remote homology detection and deep protein annotation. *BMC Bioinformatics*, 20(1), September 2019.

[12] Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35(11):1026–1028, October 2017.

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

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

[15] Pentti Kanerva. The spatter code for encoding concepts at many levels. pages 226–229, 1994.

[16] Manuel Schmuck, Luca Benini, and Abbas Rahimi. Hardware optimizations of dense binary hyperdimensional computing: Rematerialization of hypervectors, binarized bundling, and combinational associative memory. *J. Emerg. Technol. Comput. Syst.*, 15(4), oct 2019.

[17] Lulu Ge and Keshab K. Parhi. Classification using hyperdimensional computing: A review. *IEEE Circuits and Systems Magazine*, 20(2):30–47, 2020.