UNIVERSITY OF CALIFORNIA,
IRVINE

One-Class Classification with Hyperdimensional Computing

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Neftali David Watkinson Medina

Dissertation Committee:
Professor Alexandru Nicolau, Chair
Professor Alexander Veidenbaum
Professor Tony Givargis

2020

DEDICATION

To Sam and Deloris. I discover God's creation through my research, and one day you will do it with me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am grateful for professor Alexandru Nicolau's guidance. He showed me the ropes of an academic life and challenged me to discover how my ideas could change the world. Also, I have great appreciation for professors Alexander Veidenbaum, Tony Givargis, and doctor Victor Joe. They provided valuable feedback and helped transform my ideas into something tangible.

My life as a doctorate student wasn't only about research, but was also shaped by friendship. Aniket Shivam provided the support that only a lab colleague and true friend can give.

It was an absolute honor to have had the experience of working with academic staff from the University of Illinois Urbana-Champaing. Specifically with professors David Padua, Josep Torrellas, and alum Zehra Sura.

This work would not be possible if it wasn't for the sponsorship from the Fulbright-García Robles scholarship, UC Mexus and the National Council of Science and Technology (CONACYT) doctoral fellowship, the UCI Fellowship award for innovation in ICS, the UCI Latino Excellence award (LEAD), the Miguel Velez scholarship program, and the Nvidia educational grant program. Each organization provided me with key resources to start and finish this chapter of my life. Thank you.

VITA

Neftali David Watkinson Medina

EDUCATION

Doctor of Philosophy in Computer Science **2021**
University of California, Irvine *Irvine, California*

Master of Science in Computer Science **2014**
Ensenada Center for Scientific Research and Higher Education *Ensenada, Mexico*

Bachelor of Science in Software Engineering **2010**
Center for Technical and Higher Education *Ensenada, Mexico*

RESEARCH EXPERIENCE

Graduate Research Assistant **2014–2020**
University of California, Irvine *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant **2015–2020**
University of California, Irvine *Irvine, California*

Lecturer **2018–2020**
University of California, Irvine *Irvine, California*

ESL Teacher **2006–2011**
University of Baja California *Ensenada, Mexico*

INDUSTRY EXPERIENCE

Software Engineer **2010-2013**
Softtek *Ensenada Mexico*

REFEREED JOURNAL PUBLICATIONS

Differential Evolution-based Feature Selection for Sentiment Analysis **2013**
Difu100cia

REFEREED CONFERENCE PUBLICATIONS

NumbaSummarizer: A Python Library for Simplified Vectorization Reports **May 2020**
2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)

Comparison Between Online and Classroom Learning for a Boolean Algebra Class **Feb 2020**
51st ACM Technical Symposium on Computer Science Education(SIGCSE)

Evaluation of Feature Correlation for the Prediction of Sepsis **Dec 2019**
International Conference of Bioinformatics and Computational Biology (BIOCOMP'19)

Teaching Parallel Computing and Dependence Analysis with Python **May 2019**
2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS) Workshops

Using data dependence analysis and loop transformations to teach vectorization **Dec 2017**
2017 International Conference on Computational Science and Computational Intelligence (CSCI)

Using hardware counters to predict vectorization **Oct 2017**
International Workshop on Languages and Compilers for Parallel Computing (LCPC)

Survey on Remote Procedure Call Methods for Android Applications **Aug 2010**
International Conference in Computer Science (CiComp)

SOFTWARE

NumbaSummarizer `https://pypi.org/project/NumbaSummarizer/`
Python wrapper for creating vectorization reports with Numba.

ABSTRACT OF THE DISSERTATION

One-Class Classification with Hyperdimensional Computing

By

Neftali David Watkinson Medina

Doctor of Philosophy in Computer Science

University of California, Irvine, 2020

Professor Alexandru Nicolau, Chair

Contemporary research in cognitive and neurological sciences confirms that human brains perform object detection and classification by identifying membership to a single class. When observing a scene with various objects, we can quickly point out and answer queries about the object we recognize, without needing to know what the unknown objects are. Within the field of machine learning (ML), the closest algorithm that emulates this behavior is one-class classification (OCC). With this approach, models are trained using samples of a single class in order to identify membership or anomalies from query instances. However, research about OCC is scarce and most approaches focus on repurposing models that were designed for binary or multi-class classification, resulting in suboptimal performance. A novel, neuro-inspired approach to computing, called Hyperdimensional (HD) computing, promises to be closer than traditional approaches to how humans encode information. With HD computing we have the opportunity to design OCC models without having to manipulate multi-class models. This makes for a more straightforward approach that can be easily tuned to the problem requirements.

In this dissertation I present Hyperdimensional One-class classification (HD-OCC). The modeling approach uses the power of HD computing to identify anomalies among sampled data. Also, I discuss how hyperdimensional encoding works for OCC. The encoding process is

similar to those used in multi-class classification and can be reused across models.

HD-OCC is tested using three different use case scenarios. The first focuses on predicting future diagnosis of type 2 diabetes among members of the Pima Indian community. This experiment illustrates the impact of linear encoding within HD-OCC and provides a baseline comparison against ML algorithms. The second experiment uses patient data to model sepsis and predict septic shock in patients within the intensive care unit. This real-case scenario adds a different challenge in introducing sequential features to the dataset. Finally, HD-OCC is applied towards image processing by using pulmonary CT scans to detect patients with anomalies, including detecting patients with a COVID-19 infection. The results show that HD-OCC performs well and that it is versatile enough to be applied to different types of input. Also, that HD computing is a promising framework to drive research towards true artificial intelligence.

Chapter 1

Introduction

The human brain is exceptionally good at key cognitive skills necessary for knowledge acquisition. These skills are divided into 6 domains: memory, attention, learning, perception, language, and decision making [4, 65]. Researchers have tried to emulate these cognitive skills using digital systems [11]. This idea inspired the field of artificial intelligence and machine learning [73]. However, computational architectures didn't follow the same cognitive inspiration as artificial intelligence. While there's a need for a new neuro-inspired architecture, we can still emulate computational frameworks that are inspired by cognitive science. One of the proposed frameworks, inspired by the psychological memory model called associative memory [84] and the neurological processes studied by neural coding [88], is Hyperdimensional (HD) computing [60].

HD Computing is, in a sense, the product of several other ideas and frameworks on high dimensional representation. Most of core ideas behind it were first described by Kanerva [58] under the concept of sparse distributed memory, though the term HD computing was coined two decades later [60]. The logic behind HD computing is closely connected to artificial neural-net associative memories [22], and it's based on using mathematical properties of

high dimensional vectors to imitate cognitive functions with the purpose of modeling how information encoding works within the human brain (according to modern research from neurology and cognitive sciences). One of the biggest claims regarding HD computing is its closeness to human cognition and it's versatility at performing cognitive tasks. This makes HD Computing a good candidate for emulation of brain inspired computation.

1.1 Classification and the human brain

In order to identify cognitive tasks that can be implemented in HD computing, we need some understanding on how the human brain acquires knowledge. Jean Piaget was a Swiss psychologist whose main work focused on child cognitive development. He performed a series of experiments to identify key developmental milestones among young students [54]. One of these milestones was the ability to classify. Piaget concluded from his experiments that children master the ability for hierarchical classification between the ages of 7 and 10 years old. This is made evident in the way children play. Piaget observed that around that age kids start collections of toys separating them by classes (color, shape, size, evil action figures, hero action figures, and so on). This skill is essential for learning complex and abstract concepts. His ideas were echoed by contemporaries Lev Vygotsky [114], Jerome Bruner [15], among others. His work continues to inspire pedagogical and cognitive research [9, 86].

Beyond confirming that cognitive development occurs very closely to how pedagogical science models it, recent advances in neurology have discovered that the human brain is capable of performing object recognition thanks to a cascade of computational processes that amount to what is known as *Core Recognition* (or Core Object Recognition in the case of visual information) [33]. A healthy human brain can quickly recognize a known object regardless of position, size or orientation changes and with other known or unknown objects in the scene. It is still unknown the details of the underlying neural computation or how early we master

this skill, but it is evident that with enough information, we excel at discriminating known objects (Figure 1.1).This understanding about human's recognition skills is, by description, similar to how a subset of machine learning (ML) called one-class classification (OCC) works.

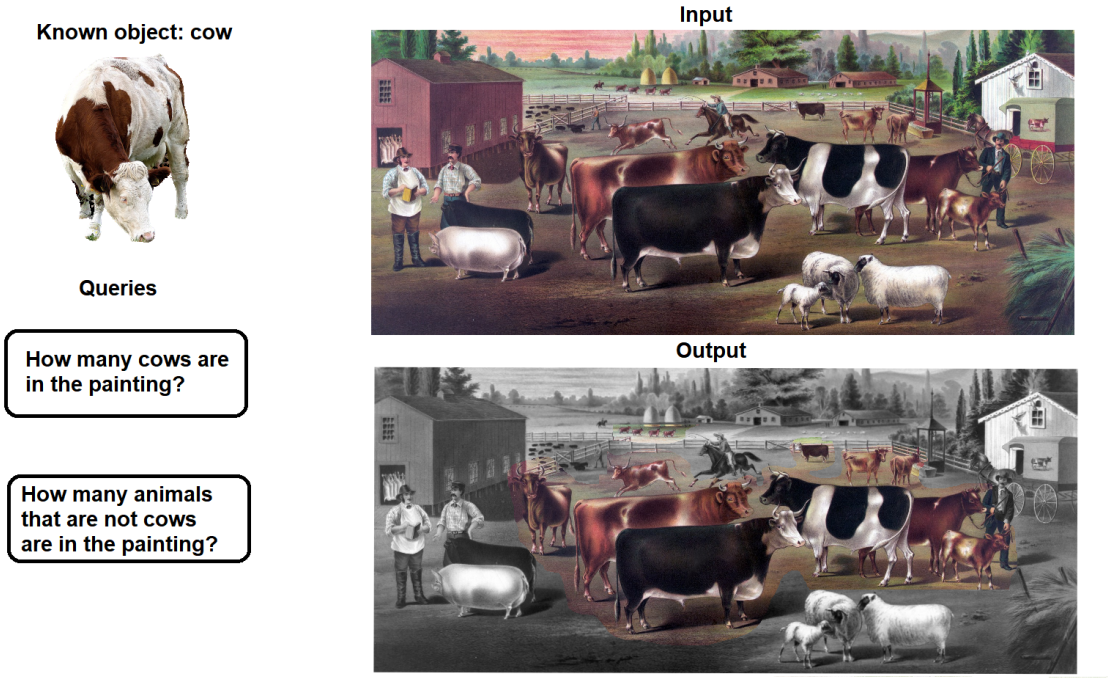

Figure 1.1: Visual abstraction of how human brains perform core object detection. If the known object is a cow, the subject is able to answer the query of how many cows are in the picture without needing to know what the other animals are. Similarly, the subject could identify objects that are not cows without necessarily differentiating the unknown object from each other.

1.2 One-class class classification

OCC, also known as unary classification or class modeling, is a machine learning approach that focuses on identifying members of a single class without modeling an opposite (or negative) one [107]. The first published paper on OCC was by Moya, et al. [79], and it is a useful approach when a problem calls for finding a specific type of instance without the need or awareness of all other types or classes represented (for example, identifying spam

documents among a body of documents from different domains) or when data about the complementary class is lacking or not available (a good use for this is detecting system failure using data that only represents a stable system). Figure 1.2 shows a representation of how an OCC model works. The model learns a single class, the *regular* class. The task is for the model to discriminate anomalies through elimination by inferring membership to the regular class. OCC is usually accompanied by a boundary threshold that determines how tolerant the model will be to instances far from the learned data.

Due to it's nature, OCC is by design closer than multi-class classification to the neurological idea of core detection [64]. Even though plenty of research is available on multi-class classification and artificial neural networks (including deep learning algorithms), there are only a few examples where neural networks are applied to OCC [96]. At the time of writing there is no method for building an artificial neural network or its derivatives (Deep Artificial Neural Network, Convolutions Neural Network, among others) for OCC. Most works modify and adapt multi-class networks to generate an OCC-like output [87]. OCC has been identified as a harder problem than multi-class classification mainly due to the challenge of tuning input parameters without looking at a sample of the opposite class [64].

1.3 Overview

In this dissertation I describe Hyperdimensional Computing-based One-class Classification (HD-OCC). At the time of writing, using OCC with HD computing hasn't been explored yet. While the overall approach is similar to how OCC works using classical ML, HD computing allows for a straightforward implementation without having to adapt a multi-class classifier. I describe three implementations of HD-OCC within the field of health informatics. Each implementation uses different concepts and tools that are common in ML but have not been used with HD computing.

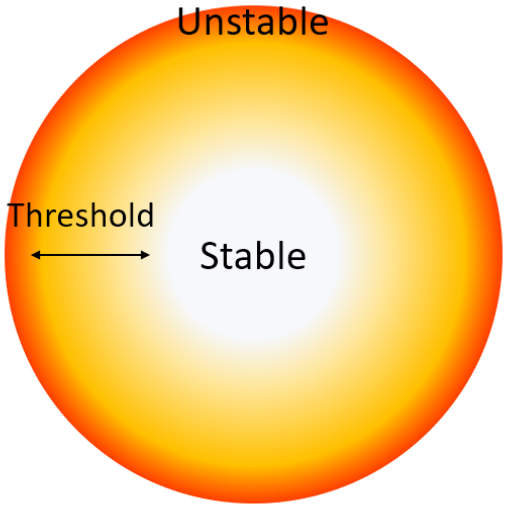

Figure 1.2: Graphical representation of OCC models applied to system stability. Data points far from the center are considered unstable according to the threshold chosen for the model.

Additionally, I explore the creation and use of synthetic data for training HD-OCC and leverage different data generation approaches. Decisions made in encoding the data have a great impact on the model's performance. This is also true to choices on how we build synthetic datasets. Since the training set is meant to be representative of all possible regular or stable conditions, a synthetic dataset should represent variability within a threshold. This threshold will be determined by the feature and guided by previous knowledge on how it correlates to the output.

The remainder of this dissertation is organized as follows.

First, Chapter 2 expands on the background and justification for this work as well as leverage the decision making for the specific implementations presented in later chapters.

Then, Chapter 3 evaluates a model using data from patients that have been diagnosed with diabetes. The model is compared to machine learning approaches for one-class classification. This work serves as a straight-forward illustration on why HD-OCC outperforms other approaches.

Afterwards, Chapter 4 describes an implementation of HD-OCC towards septic shock prediction among ICU patients with sepsis. One of the challenges in this model lies on representing sequential features with HD encoding.

Chapter 5 introduces an HD-OCC implementation for image processing. Using pulmonary CT scans from COVID-19 patients, I implemented a two step model that does one-class classification followed by binary classification.

Chapter 6 describes related work in HD Computing and One-class classification.

Finally, Chapter 7 presents the conclusion to this work and potential ideas for further work.

Chapter 2

HD Computing and Health Informatics

HD computing has been used to solve several machine learning tasks including classification [90], speech recognition [51], text analysis and clustering [60], among others. However, no published research is available regarding one-class classification and HD computing. In this chapter I present the reasoning behind the decision making on applying HD-OCC, and I explain why the implementations are within the discipline of health informatics. Additionally, a secondary major contribution is the use of synthetic data in HD computing. I explain when synthetic data is essential for HD-OCC and give a brief overview of the general approaches used for creating the data.

2.1 Health Informatics

Health informatics conerns itself with the use of information technologies for patient care [24] and most recently the use of artificial intelligence for health data processing and modeling

[92]. The field of health informatics and patient care are an optimal fit for HD-OCC because several problems within the field can be easily modeled as an anomaly detection problem. We can consider the patient state as a system's quality, and any abrupt changes as an anomaly or a system failure. In Chapter 1 I argue that OCC is akin to the human brain's ability to perform core object detection. While the problem is easily exemplified with simple object detection, the true potential for OCC can only be explored with classification challenges. The bottom line is that health informatics provides complex challenges that can be tackled by OCC.

For example, in chapter 4, I apply HD-OCC to identify patients that will go into shock among sepsis patients. Septic shock will progress differently, depending on variables such as which organs are failing. If we consider septic shock to be a systemic failure, we can define our target or training class as stable or healthy patients, depending on the problem. For every patient with an unknown state, we measure their distance to our healthy patients. The farther a patient is from the healthy samples the higher the probability of a system failure happening. However, there are important encoding decisions to be made to achieve an accurate representation of the correlation between the input and the output of the HD-OCC model.

2.2 Encoding for HD-OCC

In general, the algorithm as described by Kanerva [60] used to train for a classification task in HD computing works as follows:

1. Encode feature values into K-dimensional vectors. There are several variations on how to perform encoding. Generally speaking, within the feature domain, each value (or value range in the case of continuous data) corresponds to a unique vector. In the case

of binary vectors, the number of 1s and 0s should be equal (partially dense) in order to exploit mathematical properties of orthogonality among feature vectors (orthogonal vectors are dissimilar [40]). This means that the distance between vectors should not be dependent upon the difference in density, but about the dimensional distance between them.

2. For each data point(also known as instance or subjects, these are the units of a dataset), combine all the feature vectors to form a new k-dimensional vector. There are different approaches to do this but the most common ones are to add the feature vectors through arithmetic addition (for vectors of scalar values), multiplication (for polar vectors where every number is 1 or -1) or majority voting (for binary vector where each number is 1 or 0).

3. Depending on the application, similar vectors of the same class can be combined to form class vectors, which are hypervectors meant to represent training vectors that are similar to each other in order to save space and number of comparisons.

After generating the hyperdimensional vectors and populating the training space, for each new data point or query introduced for classification, perform inference as follows:

1. Encode the data point using the same methodology to generate a k-dimensional vector. This vector is called the *query vector*.

2. Compute the distance between the query vector and all the training vectors or class vectors. The most common distance measurement for binary vectors is Hamming distance because it is algorithmically efficient and works on a bit to bit comparison.

3. The predicted class is that of the closest training vector or class vector.

There are several variations to this algorithm including hybrid approaches where the distance metric is replaced with a deep learning algorithm. Tuning decisions would ideally be driven

by the computing architecture. For example, if the architecture allows for 5k bit operations then hypervectors of the same size are likely to be the best fit. It is beyond the scope of this research to delve in the hardware performance advantages of different parameters. For simplicity and consistency to existing research on binary HD computing, in this work I use binary 10k-dimensional (10 thousand elements) vectors and Hamming distance for prediction. Since the datasets used in the experiments described in chapters 3 through 5 are relatively small, I don't use class vectors but instead measure Hamming distances to all the training vectors for inference.

The algorithm for HD-OCC is similar to how HD multi-class classification works. The only major difference is that all the training vectors belong to one class only. Inference is not done by choosing the closest vector, but by measuring the Hamming distance between the query vector and the closest training vector. If the distance is below a predetermined distance threshold then the query vector is classified as a member of the training class, otherwise it's classified as an anomaly. This is analogous to similar threshold functions used by other OCC approaches. Chapter 3 delves into how distance thresholds work. With this in mind, a key decision for HD classification lies on the encoding algorithm being used.

2.2.1 Types of encoding

The work described in chapters 3 through 5 use three different types of encoding: orthogonal encoding, linear encoding, and n-gram encoding.

Orthogonal encoding

Orthogonal encoding is the best choice when the relationship between value of one feature is not relevant to the output. This makes orthogonal encoding ideal for dealing with symbolic

values. A perfect example of this is when applying encoding to text analysis [60]. Each letter is a symbol that holds no relevant connection to each other (A has no connection to B and B has no connection to C). In chapter 4, orthogonal encoding is applied to pixel values, as described in Kleyko, et al. [67].

The general steps for orthogonal encoding are as follows:

1. For each feature, identify the unique values (or ranges of values) that are to be encoded.
2. Generate a unique random k-dimensional vector for each value, this step would create a value-vector mapping or dictionary. This dictionary is unique for each feature.
3. After generating all the feature vectors for a single data point, they are added to form a new vector of the same dimensionality. This addition should correspond to the type of values within the vector (scalar, binary or polar) as described in the general algorithm for encoding.

A minor contribution from this work is how I implemented orthogonal encoding for high feature counts. When implementing the algorithm using a high level programming language (such as Python), an approach to keeping track of which vectors belong to which feature-values is to keep a vector dictionary. However, the size of the dictionary will be equivalent to the number of features multiplied by the number of values. A workaround is to exploit a pseudo-random number generator with the feature value as the seed to dynamically generate the vectors. This saves memory space and computational time.

Randomness

I should note that randomness is a key element for HD computing. Generating random vectors not only removes bias (forcing the model to lean towards one class or the other) but

it is the answer to brain incompatibility. In other words, just as human brains can process the same knowledge, the internal structure or neuron grouping to encode that knowledge is not the same for two brains. Therefore, it is desirable to have models start from random states. As Kanerva (2009) explains in one sentence: "If random origins can lead to compatible systems, the incompatibility of hardware ceases to be an issue" (p142). The system compatibility he is referring to is tru for both human brains and computing hardware.

Linear encoding

As stated above, using orthogonal encoding implies that we don't care about the relationship between values from a feature. However, sometimes you need to represent some kind of relationship commonly found in continuous or linear values. The implementation in chapter 3 is a good example on why linear encoding works best with patient data. In general, this type of encoding is used when the relative value of a feature contains crucial information about the instance [91]. For example, when analyzing temperature we know by intuition that a measurement of 100°F is closer to 99°F than to 97°F. This type of information can't be captured by orthogonal encoding, therefore some modifications to the encoding algorithm are needed. The linear encoding algorithm for binary vectors is as follows:

1. Similar to orthogonal encoding, identify the unique values (or ranges of values) that are to be encoded within a feature.

2. Randomly generate a partially dense vector. This is called the seed vector and corresponds to the lowest value within a feature.

3. Depending on the number of values, flip (convert 0 to 1 and 1 to 0) an equal number of bits from the seed vector to represent the next value. Repeat for the remaining values. For example, if encoding the number 1, 2 and 3 using a 10k binary vector, then the

seed vector represents 1, flip up to 10k/6 bits for 2, and 10k/6 bits more for 3. The highest vector should be orthogonal (not complementary) to the lowest vector.

4. Add all the features in the same manner as it's done with orthogonal encoding.

Step 3 states that bits are flipped in pairs. This is in order to keep the partial density of the vector.

In chapter 3 I describe a formula I used to dynamically generate vectors for linear encoding that only requires the seed vector and the indices for the 1s and 0s within it. This way we don't need to store the vectors for all possible values and we can accommodate unseen values from query instances.

N-gram encoding

Orthogonal encoding works well for discrete values that hold no relationship to each other. Linear encoding captures some of the relationship information from values within a feature. N-gram encoding is used for when the relationship is present between heavily correlated features, such as in the case of sequences. The name for this type of encoding comes from the n-gram model commonly used in information retrieval [57]. The general algorithm for binary vector n-gram encoding works as follows:

1. After encoding (using linear or orthogonal encoding) all the features, identify those that are part of a sequence and order them in descending order. For example, in a time sequence where each feature is a measurement for a different hour, then the feature with the oldest value goes first (feature 0).

2. Define the size of your n-gram. If three sequential features are joined then it will be a 3-gram.

3. Starting with the first feature (feature 0), permute (rotate the vectors by one bit) each vector n-i times where i is the feature index within the sequence and n the size of the n-gram.

4. Combine all the vectors in the n-gram using XOR. This will produce a new hypervector that will be semi-orthogonal to the individual vectors. This is key to represent the sequence as a new feature.

However, there is one issue with applying n-gram encoding to linear features. According to Kanerva [60], at high vector dimensionality, the resulting vector from n-gram encoding will be almost orthogonal to the individual vectors generated for each feature. When dealing with linear data this can be problematic. We desire for n-gram to show that there was change in value but keep the linearity of the final vector. To preserve linearity I implemented a workaround that consisted in adding the individual feature values to the n-gram using majority voting before incorporating the n-gram vector with the rest of the features. This allows for a combination of n-gram encoded vectors with linear vectors from non-sequential features. I call this approach *linear n-gram* based encoding.

The differences between n-gram and linear n-gram are better understood with an example. Figure 2.1 shows the two types of encoding being applied to a theoretical example using vectors of size 10. As noted in the image, when the sequences stay within the same value, linear encoding preserves the linearity of the original hypervectors (S2 is equidistant with S3 and S1 while S1 and S3 are the farthest pair). Figure 2.2 shows three new sequences and their distance to the previous three. The distances for the vectors generated using only n-gram encoding are inconsistent and don't represent the linearity of the values (for example S4 and S3 are close to each other even though the sequences differ considerably). With linear n-gram, sequences that are similar to each other are closer (such as S5, S4 and S2) without overlapping (with the exception of S3 to S6). The farthest pairs (S1 to S6 and S1 to S5) are almost orthogonal (distance 4) instead of complementary (n-gram distance is 8). When

dealing with patient data, linear n-gram encoding is a better fit for our model.

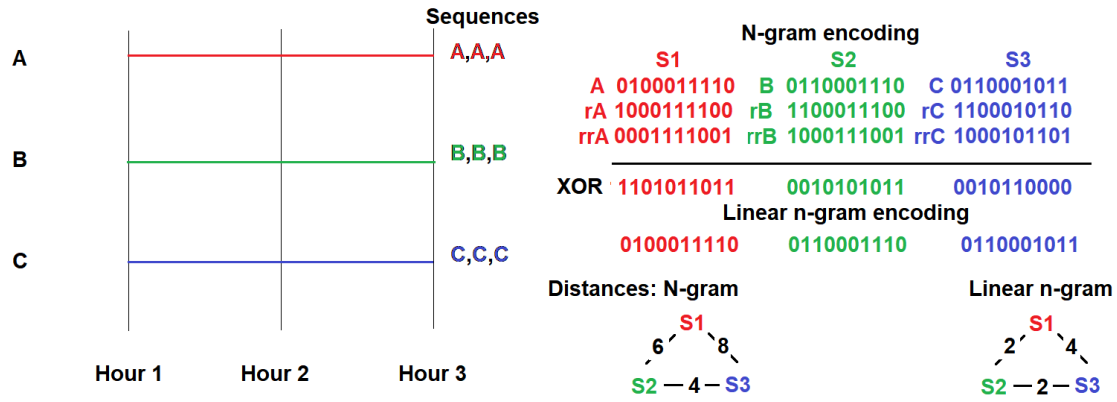

Figure 2.1: Example of the difference between n-gram encoding and linear n-gram encoding. In this figure there are three sequences that are composed of the values A, B and C. For n-gram encoding, we apply an XOR operation for the newest value in the sequence, followed by the third value rotated once (r) and the oldest value rotated twice (rr). For linear n-gram, we add the original vectors for the values in the sequence along with the n-gram that was generated.

2.2.2 Advantages and disadvantages

Encoding the known data is equivalent to the training phase in ML algorithms. HD computing doesn't depend on iterative training making it algorithmically efficient. The main advantages of HD computing in this setting are:

- Dimensionality is constant and adaptable, regardless of the number and/or magnitude of the features.
- There isn't an iterative training phase like most traditional ML algorithms do.
- The encoding algorithm relies largely on two simple math operations that can be easily parallelized:

  Addition and multiplication for numerical vectors

  Exclusive OR (XOR) and OR for binary vectors

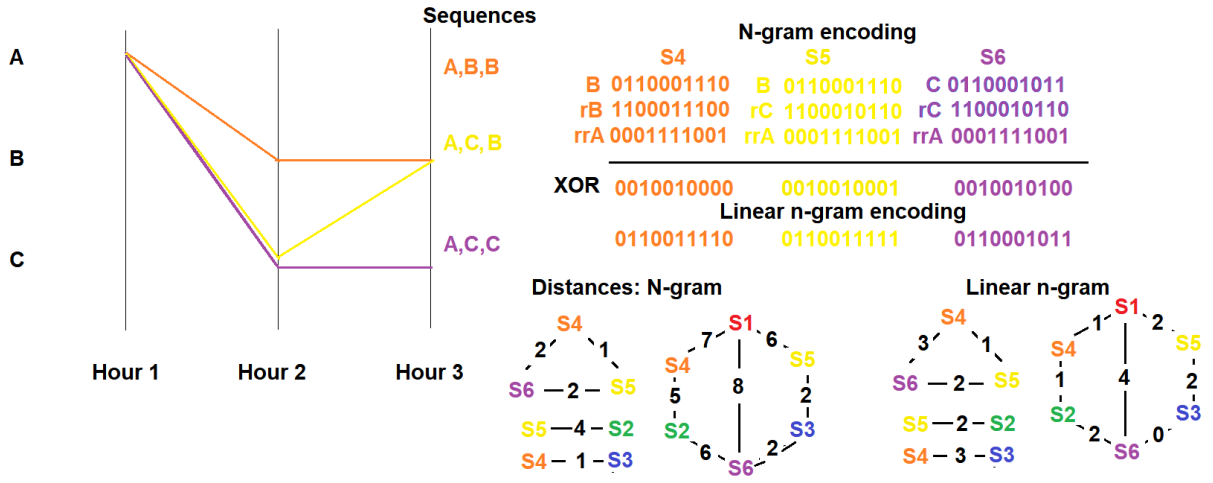

Figure 2.2: Continuing the example from figure 2.1, this figure shows three new sequences with different values. The layout for the distances was chosen for ease of interpretation. This example clearly shows that n-gram encoding introduces more volatility to the hamming distance for the vectors.

- Correlation is captured during encoding, making it highly adaptable to different domains. New data points can be incorporated without having to rebuild the model.

On the other hand, there are relevant weaknesses when compared to ML, specially when targeting a von Neumann [113, 43] architecture (chapter 6 delves into different proposals for neuro-inspired architectures). Some of the main ones are:

- The constant dimensionality can only translate to performance improvement if the hardware can perform vector operations in an efficient manner. For example, adding hypervectors is an embarrassingly parallel task.

- The main bottleneck for classical memory architectures is searching for training vectors used in computing distances. A solution could be the use of associative memories which would make search and comparison more efficient.

- Classification performance is highly susceptible to how encoding is done. This impact can only be assessed at running time, relying on trial and error.

- Depending on the application, encoding may require extensive domain knowledge to understand feature correlation. Preserving linearity or evaluating feature relevance are not part of the main algorithm and more traditional approaches are currently needed.

Computing performance issues, however, are not unique to HD Computing and emerging hardware continues to be more capable in accommodating the vector operations at the core of this paradigm.

2.3 Compiling data for OCC

The data used for training OCC algorithms needs to be representative of the system's regular or expected state. The quality of the training data is a key factor on determining the model's ability to detect anomalies. However, the data should not be overly curated since it needs to represent all possible stable values. Otherwise the model will suffer from a high number of false anomalies. On the other hand, using samples that are borderline anomalies can also be detrimental. In chapter 3 HD-OCC is trained using a sample of patients that didn't develop diabetes. However, patients that are borderline diabetic will reduce the model's sensitivity, producing a high number of false negatives. Leveraging between prioritizing true positives or true negatives will depend on the problem description.

The issue of having borderline data is made more relevant in chapter 4 where all the patients are in critical condition (they all have sepsis). For this example, there is no healthy or regular dataset. To solve this problem I created the training data using methods for synthetic data generation.

2.3.1 Synthetic data

The use of synthetic data is not new for neither ML nor OCC [72].However, this is the first time to my knowledge that it's used with HD computing. The purpose of synthetic data is to fill or create values that are missing from the real data. In the case of OCC, it can also serve as a substitution for the regular or training dataset. The main advantage of this is that we can ensure that no training instance is borderline anomalous and we can cover more possible states than we could when real data is scarce. Chapter 3 and 4 describe in more detail the specific decisions on how the synthetic data was generated. Eventually the dataset needs to be validated or replaced by real data. In the case of HD-OCC though, a synthetic dataset seems to improve the model's performance, further work is needed to understand the role synthetic data has within OCC and, separately, within HD computing.

Chapter 3

One-Class Classification for Early Detection of Underlying Diabetes in Pima Indians: A comparison Between Machine Learning Models and HD-OCC

In this chapter I explain how I adapted Hyper-dimensional One-class Classification (HD-OCC) to the problem of predicting a future diagnosis of type 2 diabetes. The target population is women from the Pima Indian community in Arizona. I discuss why HD-OCC works and compare it to other approaches to OCC, setting the baseline for works presented in the third and fourth chapter of this dissertation.

3.1 Overview of Type 2 Diabetes

Diabetes mellitus (commonly known as diabetes), is a metabolic disorder that affects a considerable percentage of the world's population, with some reports suggesting that over 8% of the world's population is living with diabetes [25]. Type 2 diabetes, the most common form (up to 90% of all cases of diabetes) is characterized by a metabolic resistance to insulin, which is a natural hormone produced by the pancreas. This results in a high blood sugar level and health complications that include heart disease, blindness, poor circulation, and death [76]. Diabetes (type 2) is treatable and preventable in most cases. Sedentarism, high body fat and poor nutrition rich in sugars and carbohydrates is heavily linked to an increase risk in type 2 diabetes being developed. However, in some cases genetic predisposition can play a key part in this probability. While individual cases might not be linked to a broader predisposition, many studies have identified key populations with a high tendency for type 2 diabetes. One of the study populations is that of the Othama People, or commonly known as Pima Indians [68].

In the early 1970's, a large study of the Pima Population sponsored by the National Institute of Diabetes and Digestive and Kidney Diseases was conducted to identify the main factors behind the high index of diabetes registered in this group. While obesity and genetics were identified as key indicators, other factors were identified as predictors too. The resulting data set from this study has been the focus of medical and machine learning research for many years. In Smith, et al.[103] an adaptive neural network called ADAP was applied to a subset of the Pima data set. They focus on predicting diabetes in women over the age of 21 that were not diagnosed with diabetes within the next year. The goal was to identify subjects who could be diagnosed with diabetes within the next 5 years.

In this chapter, I describe my experience applying HD-OCC to the Pima women data set. I argue that the algorithmic performance of HD Computing come with little sacrifice to

overall accuracy when compared to state of the art models. While HD-OCC is not directly comparable to ADAP, the first does achieve similar accuracy values while maintainging HD computing's performance gains. This work serves to argue in favor of linear encoding for medical data and the use of a synthetic data set for training. HD-OCC achieves an area under the receiver operating characteristic (ROC) area under the curve (AUC) of 0.76 when using real data for training, and 0.82 when using a synthetic dataset. The sensitivy and specificity crossover of 0.76 and 0.79 respectively. When compared with One-Class SVM, Isolation Forest, and Eliptic Envelope, HD-OCC outperforms all of them.

3.2 Modeling type 2 diabetes with HD-OCC

In order to encode the data for HD-OCC, I analyzed the data structure and how the features are correlated. This allows for me to choose the appropriate encoding algorithm. In this section I describe the dataset, its features, and the encoding I chose. HD-OCC needs a stable sample for training, so at the end of this section I explain why and how I created a synthetic data set for the task.

3.2.1 Dataset

The data captured by Knowler, et al.[68] included all adult members of the Pima community in Arizona. For a span of 5 years, they captured relevant data including family history, plasma glucose concentration, and physical measurements such as body mass index. The derived data set made initially available by Smith, et al.[103] considered additional qualifiers:

- The subject is female.
- The subject is over 21 years old.

Feature	Positive	Negative
Age	36 (21-60)	28 (21-81)
Pregnancies	4 (0-17)	3 (0-13)
Glucose	145 (78-198)	111 (56-197)
BMI	36 (23-67)	32 (18-57)
Skin Thickness	33 (7-63)	27 (7-60)
Insulin	207 (14-846)	130 (15-744)
DPF	0.6 (0.12-2.42)	0.47 (0.08-2.39)
Blood Pressure	74 (30-110)	69 (24-106)

Table 3.1: Feature distribution for the 8 features captured in the data set. The value represents the average and inside the parentheses, the range.

- Using a glucose tolerance test (GTT), diabetes was either detected within the next five years (positive) or GTT didn't detect any in five years or more.

- If diabetes occurred within one year or less of the sampling date, then the data for that subject was discarded. This is done as a curating measure to reduce the number of patients who were misdiagnosed as non-diabetic and the time of collection.

After removing subjects that had missing data, I ended up with 262 subjects in the negative class and 130 in the positive class. Table 3.1 describes the value distribution per feature. These were selected due to their relevance for diagnosing type 2 diabetes. Age and Body Mass Index (BMI) have been widely documented as correlated to type 2 diabetes [47]. Glucose and Insulin concentrations are obtained through the Plasma Glucose Concentration at 2 Hours in an oral Glucose Tolerance Test (GTT) which is a trusted source for diagnosis. Among women, number and quality of pregnancies, as well as blood pressure (diastolic) are indicators of diabetes as well [36]. Diabetes degree function and Skin Thickness were novel observations at the time of the study, though their effectiveness predicting diabetes has been documented since then [31]. In the following paragraphs I provide a brief description of these two indicators.

Skin Thickness

Triceps Skin Fold Thickness has been used as an assessment of proper nutrition since the mid-1970s [37]. The measurement is done around the upper back of the left arm. This value is often used along with BMI for assessing a patient's fat concentration. Since type 2 diabetes is heavily linked with obesity, skin thickness is as accurate as using other bodily measurements.

Diabetes Pedigree Function

DPF was developed by Smith, et al.[103] to quantify the family history with type 2 diabetes. For each subject, DPF is computed as:

$$DPF = \frac{\sum_i (K_i(88 - ADM_i) + 20)}{\sum_j (K_j(ALC_j - 14) + 50)}$$

Where i measures the ranges of all relatives who had developed diabetes before the examination date, j for all the relatives who didn't develop diabetes. K is the percentage of genes shared by the relative (0.5 for a parent or sibling, 0.25 for half sibling, grandparent or a parent's sibling, and 0.125 for cousins and parent's half siblings). Relative's age of diabetes mellitus (ADM) is the age of a relative with diabetes and Relative's age of cleared diagnosis (ACL) for a non-diabetic relative. Constants 88 and 14 are for normalizing the function according to the maximum and minimum relative ages. The constants 20 and 50 adjust the function to emphasize old relatives without diabetes and young relatives with diabetes.

3.2.2 Linear encoding

All of the features in the Pima Indian dataset follow a quasi linear correlation to the diagnosis. By this I mean that a higher value for any of the features is linked to a higher risk of diabetes. Because of this I opted for linear encoding. Recalling the linear encoding algorithm from chapter 2, the specific adaptation for this experiment works as follows:

1. For each feature, identify the lowest, *min(V)*, and highest, *max(V)* values
2. Generate a random 10k binary vector that are partially dense (has an equal amount of 1s and 0s). This will be our seed vector and used to represent every value equal or lesser than *min(V)*.
3. For all other values, flip an equal x number of 0 and 1 bits from the seed vector according to the following formula:

$$x = \frac{k(t - \min(V))}{2(\max(V) - \min(V)))}$$

   Where k is the dimensionality of the vectors, t is the target value, and V is all the values for a specific feature. The range is doubled so that the highest value gets a vector orthogonal to the vector for the lowest value.

Using the formula for dynamic creation of vectors means that we don't have to keep a vector dictionary in memory.

3.2.3 Synthetic data set

As mentioned earlier in chapter 2, having instances that are borderline anomalous will have a detrimental impact on the model's performance. To leverage this impact, I built a second

model that uses synthetic data for training instead of the real patient data. Using synthetic data for machine learning is not new [46] and it is particularly useful for imbalanced data sets and anomaly detection [72]. Using this dataset should allow for finer tuning of the HD-OCC model. At the time of writing, there's no published research with synthetic data being used for HD computing.

The synthetic data is meant to represent a subject with the lowest probability of being diagnosed with diabetes but still accounting for variations in data. Each synthetic feature was built as follows:

- **Age**: All the synthetic patients have 21 years old as age. This means that patients with age higher than 21 will be further away than patients that are equal.
- **Pregnancies**: According to related studies, having 4 or more pregnancies increases the observed probability for diabetes. For each synthetic subject I generate a random number in the range of 0-3.
- **Glucose, BMI, Insulin and Blood Pressure**: The numbers for these features are randomly generated within the *normal* or *healthy* ranges as published by the Centers of Disease Control and Prevention (CDC). These ranges are: 80–100 mg/dL for glucose, 18–25 kg/m^2 for BMI, 0-100 mIU/L for insulin, and 60-80 mm Hg for diastolic blood pressure [61].
- **Skin Thickness**: In women, healthy skin thickness should be less than 18 mm with 25 mm being the threshold for obesity. The values for the synthetic database range from 1 to 20 mm.
- **DPF**: This value was created to serve as a potential predictor for diabetes. Since relevant information is already encoded in it and every value higher than 0 carries a risk for diabetes, the synthetic database has a constant value of 0.

The synthetic data set was populated with 50 subjects using these generation approaches. This number was chosen arbitrarily and was guided by the variables deviation. Adding more synthetic subjects didn't increase the model's accuracy.

One of the models will be trained with only synthetic data. While real and synthetic data can be combined for better performance, the purpose of the model is to highlight the difference between using synthetic and real data.

3.3 Implementation

The two models (one built with real data for training and the other using synthetic data) were tested. Due to the different data origins the testing was done according to what the model required.

3.3.1 Model using real data

For the first model, I build the training vector space as follows:

1. From the negative class, randomly extract 75% for training and keep the remaining 25% for validation
2. Encode each subject using linear encoding.
3. Store each vector in memory. If an associative memory is being used, we can eliminate vectors that are similar to each other to eliminate redundancy.

For validation, we encode each patient from the validation set and form the positive class set. I then measure the Hamming distance between the incoming vector and the vectors in the training set.

3.3.2 Model using the synthetic data set

The steps for training with synthetic data are very similar to the first model. However, only synthetic data is used for training and all the negative and positive datasets are used for validation. It is important to note that I only generated 50 synthetic subjects, compared to the 200 subjects used in the real data training. While there's no rule regarding how many synthetic subjects we should generate, building a large set will introduce redundancy and diminish the advantages of using synthetic data.

3.3.3 Results

The model reported in Smith et. al.[103] for binary classification reports an Area Under the Operating Curve (AUC) of 0.72 and a crossover value for sensitivity and specificity of 0.76. Table 3.2 shows the performance values for HD-OCC. While the two approaches are not directly comparable, it is interesting to find that HD-OCC achieves similar performance as the binary model that uses a computing intensive algorithm (ADAP).

The synthetic model clearly outperforms the one using real data. One of the main reasons behind it is that the synthetic data set is *cleaner*. There are no borderline cases and the impact from features, such as number of pregnancies or DPF, are amplified.

3.4 Comparing to classical machine learning

One-class classification has proven challenging for deep learning [96]. Solutions using big data algorithms are mostly tailored for the domain. Since this work is focused on HD computing applied to OCC, it is beyond our scope to build a new custom algorithm that uses ML in order to compare with HD-OCC. Instead, I turn to classical machine learning approaches

	Training with real data	Training with synthetic data
AUC	0.76	0.82
Crossover	0.72	0.74
Specificity at different sensitivity thresholds		
Sensitivity		
0.90	0.37	0.61
0.70	0.73	0.77
0.50	0.82	0.88

Table 3.2: Area Under the Operating Curve (AUC), sensitivity and specificity crossover, and specificity values at sensitivity intervals 0.9, 0.7 and 0.5 for both the model built with real data and the one with synthetic data.

for one-class classification. I built three models for the comparison using: One-class SVM, Isolation Forest and Elliptic Envelope.

3.4.1 One-Class SVM

This algorithm is a unary version of the Support Vector Machine algorithm for binary and multi-class classification. Given samples of a single class, One-Class SVM learns the boundary of the *normal* class. Then, new instances are determined to be either inside the boundary, therefore members of the regular class, or outliers, for which they would be categorized as anomalies, or for the purposes of this experiment, diabetic (see Figure 3.1). One-class SVM has been used in the field of image recognition, fraud detection, health informatics, among others[20]. The two main tuning variables are γ, and ν which control how far and how smooth the boundaries are, with ν reflecting the proportion of anomalies that we expect to see. Table 3.3 shows the best accuracy achieved with γ=0.00001 and ν=0.7.

3.4.2 Isolation Forest

Following a tree-like structure, Isolation Forest randomly selects features and creates a split between the maximum and minimum number. The hypotheses is that *normal* observations

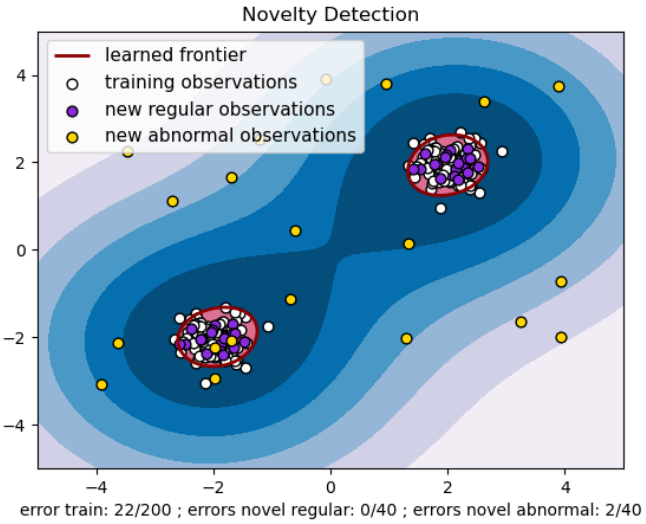

Figure 3.1: Representation of how One-Class SVM works. The purple dots, which are the *normal* observations, should optimally fall inside the boundary created by learning from the white dots (training data). The yellow dots are the anomalies, or in the case of this project, people with diabetes. Generated using code from `https://scikit-learn.org/`

will take a higher number of splits to be isolated from other samples than it would for anomalies. Therefore, samples belonging to the regular class will tend to be closer to the classification space and anomalies will be scattered, isolated (see Figure 3.2. The splitting tolerance is controlled by a contamination ratio, which is determined by the expected proportion of anomalies in new observations [71]. This is akin to the ν variable from One-Class SVM. The best accuracy was achieved with a contamination ratio of 0.7. Higher ratios diminish the resulting AUC.

3.4.3 Elliptic Envelope

Elliptic envelope assumes that the data follows a Gaussian distribution and fits an imaginary ellipse around the training observations. All the new observations that fall outside the boundaries of the ellipse are treated as anomalies (see Figure 3.3).

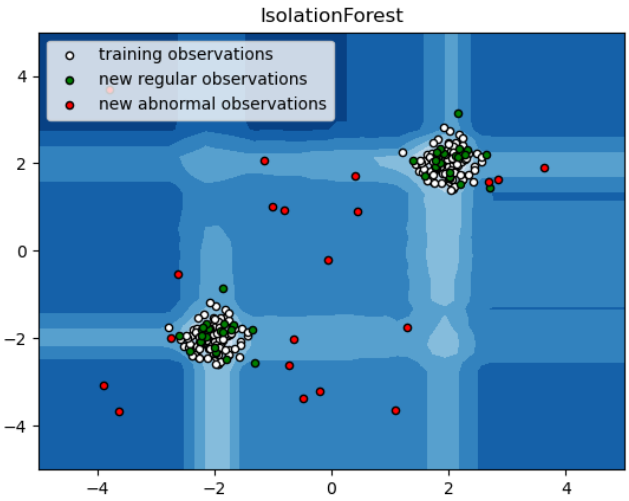

Figure 3.2: Graphic representation of Isolation Forest modeling. The goal of the algorithm is to isolate anomalies by creating paths where observations similar to the training samples will be together and those that are not similar will be left out.Generated using code from `https://scikit-learn.org/`

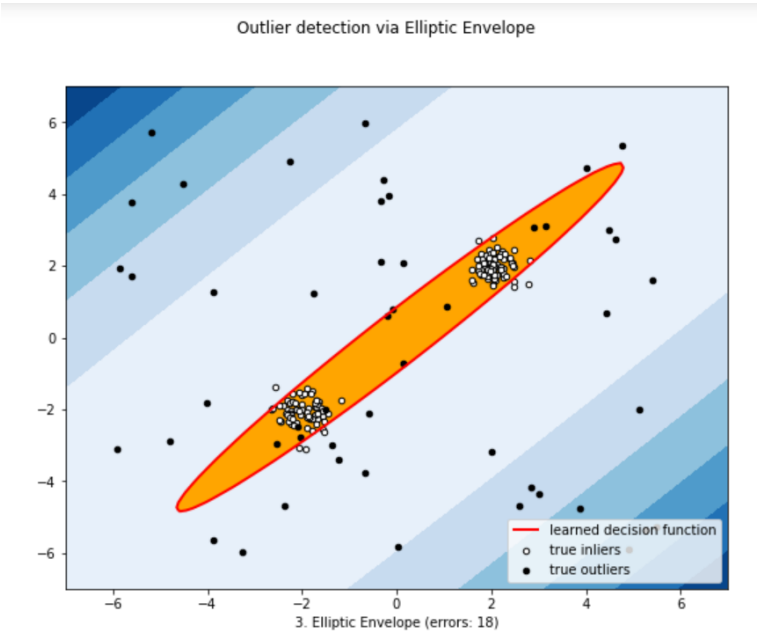

Figure 3.3: Elliptic envelope where the data, though clustered in two separate spaces, is surrounded by a single ellipse. The boundary can be relaxed using a contamination ratio similar to how Isolation Forest is tuned.

Model	Accuracy	AUC
One-class SVM	0.68	0.40
Isolation Forest	0.70	0.42
Elliptic Envelope	0.74	0.38

Table 3.3: Accuracy and AUC values for traditional machine learning algorithms for One-class Classification

3.4.4 Comparing with HD-OCC

The boundary for HD-OCC is controlled by the distance threshold. For 10k vectors, the distance threshold can be any number between 0 to 10,000. The higher the number the more tolerant the model will be towards longer distances being declared as regular. At the moment there is no detailed guideline on the optimal value for this threshold. One option is to compute distances between all training vectors and set the threshold to be higher than the highest distance observed.

While further tuning could be done for both HD-OCC and the classic ML models used for this experiment, the first outperforms ML classifiers. This is due, in part, to HD-OCC fitting every sample in the training data. This could be detrimental if the training set is noisy since it could tend to overfitting. However, that tendency can be countered by using class-vectors instead. For this project I decided against class vectors because of how small the data set is. However, as the amount data grows, those two aspects of HD-OCC can be balanced.

Another reason for why HD-OCC outperformed ML is because HD-OCC allows for feature encoding customization. This is more evident in the case of the synthetic data set. The decision to allow age and DPF gain relevance on determining the observation's distance was guided by previous knowledge on the feature's correlation to the output (by setting those features to constant values in the synthetic data set). The more extensive this knowledge is, the more accurate the encoding will be. This property is evident in HD computing in general but more evident in HD-OCC due to the assumption that anomalies are not common

and therefore decisions on how to encode for them are based on intuition and experience. While this can easily be argued as a drawback, it is not unusual to have finely tuned feature creation for machine learning [104, 95]. Several techniques for feature creation such as bag-of-words started as domain-specific approaches that are now finding general usefulness in other algorithms [115].

3.5 Summary

In this chapter I presented an HD-OCC model for type 2 diabetes detection among women in the Pima Indian tribe. The model outperforms classical approaches to One-class classification and has similar accuracy performance to neural network based algorithms. The best model was achieved using a synthetisized dataset built to represent subjects with minimal probability of being diagnosed with diabetes. This model achieves 0.82 AUC with a sensitivity/specificity crossover of 0.76. I also reason that HD-OCC is extremely customizable and can fit feature correlation in the encoding, though further exploration is needed to find a standard feature encoding framework. While linear encoding was the obvious choice for the model presented in this chapter, further work is needed to establish design guidelines on synthetic data creation for HD computing, as well as choosing the right metrics such as the distance threshold. In the next chapter a similar approach is used for patient data but with the added challenge of encoding sequential features.

Chapter 4

Hyper-dimensional Class Modeling of Septic Shock Among Sepsis Patients

In this chapter I describe HD-OCC applied towards the detection of septic shock among patients with sepsis. This introduces two main challenges different from the model presented in chapter 3. These are: the representation of sequential features, and the complete lack of a healthy patient sample for training.

A proposed use for this application is for it to become the backbone of an embedded alarm system that predicts hypotension (the main indicator of septic shock) before it happens. The model targets septic patients in an ICU setting.

4.1 Sepsis, Septic Shock, and Scoring Systems

When a patient is infected by a disease causing organism, the immune system generally responds to it through different defense mechanisms. In some circumstances, the immune system can have an extreme response to the infection. This reaction may cause severe tissue

damage to organs and eventually, death. The term for referring to this condition is sepsis, and it is the leading cause of hospital deaths in the United States (1 in every 5 deaths) [75]. However, with early detection sepsis can be managed with antibiotics (when the source of infection is bacterial). Survival rate is over 70% if properly diagnosed. If sepsis goes unnoticed, or the the infection is resisting treatment, a patient will suffer multiple organ dysfunction that causes a systemic shutdown. This phase is called septic shock and the patient is considered to be in a critical state. Once shock starts, the survival rate goes down to 20% [30].

Current protocol [99] defines the clinical criteria for detecting septic shock as the presence of hypotension that won't respond to fluid resuscitation, accompanied by associated tissue hypoperfusion, and requiring the use of vasopressors. However, there are different proposed guidelines regarding on when vasopressor usage should begin. The earliest suggested starting time being simultaneous to fluid resuscitation in response to hypotension [5, 69].

The definition of sepsis and its subsets (severe sepsis and septic shock) has evolved overtime. The reason behind this is partly due to the lack of an accurate criteria or biomarker for detecting sepsis and its different phases. Added to this problem is the fact that hospitals follow different protocols and scoring systems to determine if a patient is septic [34]. The four most common systems used are:

- SIRS (Systemic Inflammatory Response Syndrome): The oldest in this list and until 2016, it was the leading criteria model for defining sepsis. SIRS uses a simple binary scoring system with 4 variables related to: Temperature, blood pressure, white blood cell count, and heart rate [28]. If each value is outside a defined threshold then the score is incremented by 1. SIRS is still utilized by many hospitals. Several studies, however, found that SIRS tends be oversensitive (high number of false positives) and not reliable for guiding treatment [110].

- MEWS (Modified Early Warning Score): Most commonly used in the United Kingdom, comprises 6 vital signs and assigns a score from 0 to 3 to each one according to predetermined thresholds. A cumulative score of 4 or more triggers a call to a Rapid Response Team. MEWS is the only scoring system in this list that has a specific response protocol attached to it. However, MEWS was not specifically designed for sepsis so accuracy in identifying this condition is low [106].

- SOFA (Sequential Organ-Failure Assessment): Previously known as the Sepsis-related Organ-Failure Assessment. Since 2016, SOFA is part of the official Sepsis-3 definition [101], that defines sepsis clinically as an acute increase of 2 or more in SOFA score. This scoring system is similar to MEWS in the sense that it uses a cumulative score over a number of vital signs (6 groups in SOFA's case). The primary objective for SOFA is to identify organ failure. Studies have found that SOFA has higher sensitivity and specificity than SIRS or MEWS, but it is considered to be the most burdensome and impractical for quick response or continuous monitoring scenarios. Quick-SOFA (or qSOFA) is a version that uses a shorter criteria (3 values). However, qSOFA requires a pre-established suspicion of sepsis, and it's meant to be a monitoring tool in non-ICU (Intensive Care Unit) settings. While SOFA is generally used as a one-time assessment, studies have suggested using SIRS for sepsis detection and qSOFA as a triage tool [112, 34].

Some works have tried to produce a predictive tool that would help in defining a protocol for septic shock [66, 41]. A common approach is to use the mentioned scoring systems as a guideline for these tools. The effectiveness in modeling septic shock has been limited [35].

4.2 Encoding patient data

The primary goal for this illustrative example is to create a system that can predict septic shock before it happens. I are targeting ICU patients with confirmed sepsis because I want the model to predict their outcome and potential for going into shock. I identify our targeted event as the time when a patient's mean arterial pressure (MAP) goes below 65 mmHg (hypotension) and/or lactate levels are over 2 mmol/L (tissue hypoperfusion). These are the two main indicatives or preambles of septic shock.

4.2.1 Linear encoding

As demonstrated in chapter 3, patient data is linear by nature, and the relation between feature values is relevant for the output. For this data I can use the same linear encoding. However, this time I also have sequential values. Encoding each value in a sequence using linear encoding and adding them to the patient vector will result in a higher relevance given to these features. This is because the vectors for each value in a sequence is expected to be relatively similar to each other. When added to the patient vector using majority voting they will have a bigger *share* in the voting process. Therefore, I need to use a different encoding approach for these features.

4.2.2 N-gram encoding

A suggested approach for dealing with sequences is to use n-gram encoding. As explained in chapter 2, this approach was first used for text analysis and is analogous to how the classical n-gram model works. To show how the generic algorithm described in chapter 2 applies specifically to this dataset, the steps I followed are:

1. Given a sequential feature with $n$values, encode each value using linear encoding. All the sequential values for the same feature will use the same seed vector and linear ranges.

2. Starting with the last (newest) value in the sequence, rotate (transpose) the corresponding vector i times to the left, where i is the index of the value relative to the last one (starting from 0). For example, the second to last value will be rotated 1 time, the third to last would be rotated 2 times, and so on.

3. Combine all the rotated vectors using XOR.

While n-gram has been used in other works within HD computing, most of them have a single sequential feature. In this case I am using a mixture of sequential and linear features. Because of this, I opted for linear n-gram based encoding as described in chapter 2. This way, if there's little change across the sequence, the n-gram vector will have little impact on the final vector. If there's an abrupt change then the n-gram will be made evident after majority voting, with a higher chance of making the final vector anomalous or unstable.

4.2.3 Dataset

One of the most popular databases for health informatics research is MIMIC-III [56]. However, MIMIC-III uses an outdated definition of sepsis and its subsets. This has great impact in how the class is represented. Instead, I use eICU [89], a newer database from the same team behind MIMIC-III that uses the updated definition of sepsis.

Since eICU has data from patients within the ICU, I don't have access to data from healthy patients. On the other hand, the problem I are trying to solve is to detect patients with confirmed sepsis that are at risk of going into shock. Due to data constraints, I will focus on bacterial infection related sepsis. I don't consider fungal or virus related sepsis due to

Feature	Positive	Negative
Age	63 (17-89)	61 (18-89)
WBC	11 (0.1-42)	10 (0.6-46)
MAP	83 (65-144)	94 (65-147)

Table 4.1: Distribution for three relevant features within the positive (septic shock) and negative (not septic shock) classes. The value represents the average and inside the parentheses, the range.

lack of samples. I also omit sepsis in burn and cancer patients due to different treatment interactions and symptom overlap between the two.

The general population criteria is: patients admitted into the ICU with confirmed sepsis (using Sepsis-3 criteria) who were not in shock when admitted, underwent antibiotic treatment, and have at least three hours worth of data. Using this criteria, I obtained data from 1237 patients before applying the data filters for the positive and negative classes. Table 4.1 displays the value distribution of three relevant features for each class.

4.2.4 Positive class

Before trying HD-OCC, I built a baseline model that performs binary HD classification. Within the context of binary classification, the positive is built with patients who, in addition to our population definition, underwent sepsis related hypotension (MAP¡65 mmHG) and/or hypoperfusion (lactate¿2 mmol/L), and were not being treated with vasopressors previous to this event (387 patients). The timestamp for this event is denominated as Hour Zero (H0).

4.2.5 Negative class

The complementary, or negative class, are patients from the defined population who didn't have a shock related event during their stay, were not treated with vasopressors, and were released from the hospital without dying (567 patients).

Features

We consider two types of features:

- Sequential or periodic: These are values that change overtime and are available to build sequences. I consider 4 sequential features:
  - Mean Arterial Pressure (MAP)
  - Respiration Rate
  - Oxygen Saturation (SaO2)
  - Heartrate
- Static: These are features that don't change within the observation window, I consider 6 features:
  - White Blood Cell Count (WBC)
  - Red Cell Distribution Width (RDW)
  - Blood Urea Nitrogen (BUN)
  - Creatinine
  - Sodium
  - Age

The criteria for feature selection is defined by availability of the data, and relevance within related work that point towards a correlation between the feature and sepsis / septic shock. For example, temperature is a feature that is highly correlated with sepsis and infection in general, but surprisingly, it isn't widely available for all patients in the database.

For the positive class, sequential features are collected at 1, 2 and 3 hours before H0, considering the hourly average. For the negative class, collected features are from 1, 2 and 3 hours

after sepsis is confirmed and antibiotic treatment has started, or after admittance, whichever is later.

Synthetic data set

Since all the population in our data is from septic ICU patients, I don't have a reliable baseline for HD-OCC. Instead I created a synthetic data set with feature values that are considered as *healthy* by current medical standards. The methodology chosen to generate synthetic patients is similar to the one used in chapter 3 but adjusted to the new features. I created synthetic patients with each value randomly assigned within a *healthy* range. Then the data is encoded using the same methodology used for the real patients.

4.3 Implementation

In this section I describe the results using eICU. The general approach is meant to be adaptable to other data sources and would ideally get live input from a vital signs monitor and from the electronic health record.

4.3.1 Binary Classification

The following models follow the classical HD Computing algorithm for binary classification as presented in [60]. The purpose of these models is to understand the relationship between the positive and negative classes.

Hour	**True Positives**	**True Negatives**	False **Positives**	**False Negatives**	**Overall Accuracy**
-3	214	394	173	173	0.64
-2	188	397	170	199	0.61
-1	217	412	155	170	0.66
2-gram (-3+-2)	201	397	170	186	0.63
3-gram	208	406	161	179	0.64

Table 4.2: This table shows accuracy numbers for the HD Computing model using binary classification. Hours are in the perspective of the positive class, -3 hours means 3 hours before hypotension is registered. For the negative class this would be the first hour of observation.

Raw value encoding

I built three models that use the static features plus sequential features for each respective hour. Table 4.2 shows that there's a slight increase in accuracy from the majority class (in this case, the negative class) for every hour. The last hour has the highest overall accuracy but all three models are similarly distributed.

N-gram encoding

Table 4.2 also shows how using n-gram encoding for sequential features performs. It is interesting that n-gram models seem to have similar accuracy numbers as the models that use raw values. One thing to consider is that the sequential values are coming from hourly averages which might have a smoothing effect on the generated vectors.

4.3.2 HD-OCC Model

In Table 4.3, I describe the specificity at different sensitivity thresholds for HD-OCC. This is a more descriptive perspective than reporting overall accuracy since the strength of this approach is in its adaptability of prioritizing the detection of positive cases or discarding false

positives. While specificity seems low at high sensitivity thresholds, I must consider that the negative and positive classes are very similar before a patient goes into shock. Furthermore, replacing the synthetic dataset with real data or with a different generation approach to synthetic patients, could improve these numbers.

Age multiplier

After further examination, I realized that age was having a detrimental effect on the model's accuracy. A possible explanation is that while age is not necessarily an indicator of sepsis, it has a semi-linear correlation to poor outcomes. To represent the impact that age has with sepsis patients, I decided to not add age to the final vector through majority voting, but instead multiply the final Hamming distance by an age ratio (age divided by 40). This is in accordance with [17] that state that for patients older than 50 years old, the probability of septic shock increases. Therefore, distances for patients with an age close to 40 will stay overall the same, whereas higher ages will increase the final distance and lower ages will decrease them. While this approach remains to be validated, it did improve accuracy considerably.

Table 4.4 shows the specificity numbers for the model with age as a multiplier. Most are an improvement over the previous model that encodes age as part of the patient vector.

To place this model into perspective, the best model for binary classification (-1 hour) has a sensitivity of 0.58 and specificity of 0.69. The HD-OCC model has similar numbers at that threshold, but with the added benefit of being adjustable to prioritize sensitivity (reduce false negatives) or specificity (reduce false positives).

Model	Specificity at sensitivity:		
	0.9	**0.7**	**0.5**
-3 hours	0.16	0.45	0.61
-2 hours	0.16	0.47	0.66
-1 hour	0.19	0.51	0.66
2-gram (-3+-2)	0.19	0.46	0.64
3-gram	0.19	0.48	0.65

Table 4.3: Specificity numbers for OCC model using a synthetic dataset. The numbers for specificity represent specificity at sensitivity intervals of 0.9, 0.7 and 0.5 respectively

Model	Specificity at sensitivity:		
	0.9	**0.7**	**0.5**
-3 hours	0.17	0.46	0.63
-2 hours	0.18	0.48	0.69
-1 hour	0.21	0.53	0.69
2-gram (-3+-2)	0.19	0.46	0.66
3-gram	0.19	0.50	0.68

Table 4.4: Specificity numbers for OCC model multiplying distance by age. The numbers for specificity represent specificity at sensitivity intervals of 0.9, 0.7 and 0.5 respectively

4.4 Discussion

HD-OCC performed well with the patient data though not with the same accuracy numbers as the models in chapter 3. A major obstacle is inconsistency within the data. There are features such as temperature that were missing from most patients. For the few that had it, the value was measured outside the observation window, most often coming from a different hospital unit. Furthermore, this inconsistency has a negative effect on the granularity of the observation window. The data for the sequential features is continuous, however it was registered at different intervals. Because of this, I used hourly average instead, but this decision doesn't allow us to use the full potential of n-gram encoding.

With more control over data input, such as using live data through an embedded solution, the models are bound to improve. Specially with false positives. N-gram encoding amplifies the distance to the synthetic dataset when there are abrupt changes in the information, but averaging sequential data has a smoothing effect. A patient who has low blood pressure, for example, but is not going into shock, will have a lower distance than one with optimal values that rapidly changes over time.

Feature cost

Feature cost refers to the effort (time, work, or in this specific case, intrusiveness of data collecting methods) needed to obtain it. An expensive or costly feature is one that requires more effort than the average or cheap features. For example, our model uses 4 features that are captured from the vital signs monitor.This monitor is commonly found as a bedside monitoring tool and once connected to the patient, commonly with supra-cutaneous diodes, measurements don't require any intervention. In this context the features are considered to be cheap features. On the other hand, WBC, Creatinine, Sodium and RDW are more expensive because they require drawing blood from the patient and send it to a lab for

Feature Group	SIRS	MEWS	SOFA	qSOFA	HD-OCC
Temperature	Yes	Yes	No	No	No (Pending)
Heart Rate	Yes	Yes	No	No	Yes
Respiration Rate	Yes	Yes	Yes	Yes	Yes
General Blood Test	Yes (WBC)	No	Yes	No	Yes
Response evaluation	No	Yes (AVPU)	Yes (GCS)	Yes	No
Blood Pressure	No	Yes	Yes	Yes	Yes
Liver	No	No	Yes	No	No
Urine	No	Yes	Yes	No	No

Table 4.5: Feature table comparing our approach (HD-OCC) with industry standards. The first column is named Feature Group because the features are not exactly the same across models but collection methodology is similar. Response evaluation encompasses different types of tests performed by medical staff to quantify the patient's level of consciousness. GCS stands for Glasgow Comma Scale and AVPU stands for Unresponsive, Responds to Pain, Responds to Voice, and Alert.

analysis. However, as expensive as these features are, they are part of the same, standard protocol, blood screening. Added to this, the model's backbone, that is the sequential data, come from cheap features. Having new expensive data added can help the model's accuracy but is not critical to the model's performance.

Another type of expensive feature are those that require direct medical intervention or assessment and are mostly qualitative in nature. In this sense I can compare our model to SOFA, since the later relies on this type of features. As shown in Table 4.5,

MEWS and SOFA are expensive models in terms of feature cost. SIRS and our model are cheaper and don't rely on a medical assessment.

Flexibility

Hospitals don't have the same protocols and the same data available every time. Sometimes the main challenge for adopting a new criteria is lack of data needed for it to work. In terms of flexibility, our approach can include any data that is readily available while MEWS and SOFA have a strict protocol attached to them. However, I consider the 4 sequential features

as being necessary since they are readily available and capture changes in patient state with higher precision than other features.

4.4.1 Strengths and weaknesses of our approach

Unlike classical ML, HD-OCC allows us to use intuition and expertise to improve the model and make it robust. This model falls under a more specialized and tuned up approach that is adaptable to domain constraints.

Compared to industry standards, HD-OCC offers greater promise as an ever growing and always improving model. The more patients are treated, and more data is made available, the more opportunities for the model to improve and be finely tuned. This is a desirable property from an artificial intelligence approach.

This is not to say that there aren't any drawbacks. There isn't a proven methodology for feature evaluation within HD Computing. This means that new features need to be carefully evaluated with more conventional techniques since I have little insight on how they interact with the final class once added to the patient vector. HD Computing requires domain knowledge and this is true also if the feature space is being expanded or the model is applied to a different target that is not septic shock.

Another issue is how descriptive the model is. HD computing is easy to understand mathematically. I can even derive why a feature value increases or decreases distances to a base class. In this sense it does a better job than most ML approaches at providing an explainable model. But I currently can't provide an automatic explanation to why a specific combination of feature values is considered far or close enough, nor can we, if possible, translate the model into a non-dynamic criteria.

4.4.2 Work related to Septic Shock modeling

The work of Giannini, et al.[41] solves a similar problem to the one presented in this chapter. However, they target patients outside the ICU. While the feature burden is equivalent to ours, they follow an old definition of sepsis. Furthermore, their model prioritizes specificity (over 90%) while sacrificing sensitivity (27%) without a way to modify the model.

The work of Kim, et al.[66] uses an ML approach to model septic shock. In this work they include chief patient complaint as a feature and encode it into numerical values. While their sensitivity (0.70) and specificity (0.90) are considerably higher than our approach, their study is retroactive to septic shock, with the observation window including patients that underwent shock already. Furthermore, their positive class is comprised of patients between 66 and 88 years old and the negative class age range is 51 to 79. Due to the little overlap between classes, they confirm age being a dominating feature in their model.

The challenge described in this chapter, however, is unique in the way the classes are defined. In the case of binary classification, the positive and negative classes are not linearly separable by any of the features and are similar to each other from the point of view of patient state. I target a critical observation window ensuring that patients have not gone through septic shock before making a prediction.

4.5 Summary

In this chapter I described the application of HD-OCC to the prediction of septic shock among sepsis patients. The model performs consistently across use cases achieving up to 66% classification accuracy when balancing specificity and sensitivity but can be adjusted to prioritize either. I argue that application of this model doesn't pose an extra burden on current protocol and can provide an early alarm for preparation or prevention of septic

shock. The major contributions of this work to the idea of HD-OCC are:

- Using linear N-gram encoding to represent sequential features
- Using a synthetic dataset as the baseline for training HD-OCC and make predictions on real data
- Propose an application of HD-OCC that involves real time data and instant feedback
- Adjust final distances based on feature relevance. In this specific case, use age to adjust distances for improved accuracy.

Chapter 5

Detecting COVID-19 Related Pneumonia in CT Scans using HD-OCC

This chapter delves into HD-OCC used towards image processing. Unlike the models from chapter 3 and 4, doing image encoding implies a reduction of dimensionality. In this case, we have 60 thousand features (pixels) and the final representation has the magnitude of 10 thousand bits. The goal is to detect pulmonary diseases from a single CT scan slice.

5.1 Overview of COVID-19

On January 2020, the World Health Organization declared a global emergency due to a novel coronavirus that had started in the regions of Wuhan, China and rapidly spread around the world. Symptoms include fever, headache, loss of smell, difficulty breathing, among others [18]. In the early stages of creating a diagnosis protocol, medical experts were specifically

interested in the effects SARS-CoV-2 (COVID-19) had in the patient's lungs. In severe cases, lungs get inflamed, filled with fluid and debris, causing what is known as pneumonia [85]. Before any quick test was developed to identify the infection, hospitals relied on computerized tomography (CT) scans [50] and using lower respiratory tract samples [117]. However, accuracy among expert radiologist on differentiating COVID-related pneumonia from typical pneumonia can vary from 97% down to 67% [7].

At the time of writing, various tests have been developed, but their accuracy has been the focus of debate. While most tests seem to emphasize sensitivity over specificity [19], CT scans remain the most accurate way to confirm symptomatic infections, especially when dealing with in-hospital settings and with patients with preexisting lung related conditions [116].

Recent research has focused on applying artificial intelligence to the challenge of detecting COVID-19 in CT scans [70]. The work of Soares, et al.[105] aims to build an explainable deep learning network using CT scans collected from patients across several hospitals in Sao Paulo, Brazil. They released their dataset containing images from healthy patients, patients with COVID-19 related pneumonia, and patients with other pulmonary issues. For this implementation of HD-OCC I use random vector generation for encoding pixel values from the mentioned pulmonary CT scans. I first apply HD-OCC for anomaly detection among all CT-scans and then HD computing based binary classification to differentiate COVID-19 scans from non-COVID-19 illness scans.

5.2 Methodology

The data set has 46 scans from healthy patients, 80 scans from patients diagnosed with COVID-19, and 67 from patients with non-COVID-19 pulmonary illnesses. Imaging data

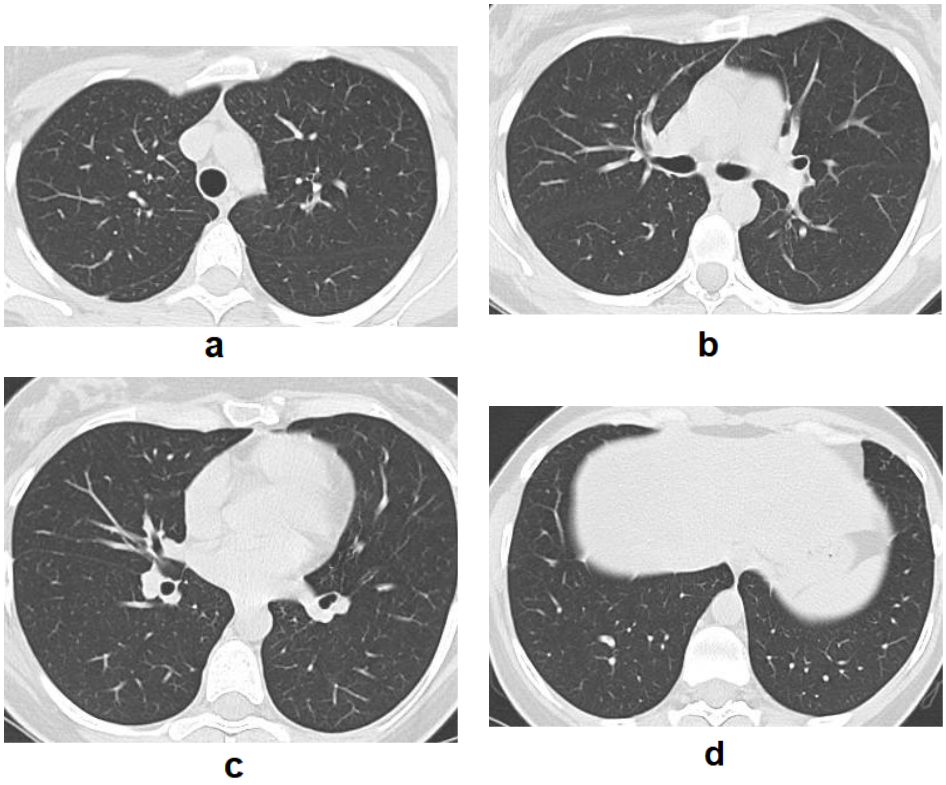

Figure 5.1: CT scan image sample from one patient showing the upper lobe and the trachea in the middle of the image(a), the mid section of the lung showing the anterior segment in focus and the trachea splitting into the main bronchi (b), the lower lobe section with inferior lobar bronchi (c), and the basal segments of the lower lobe with the diaphragm starting to appear (d)

from CT scans are generally stored using the DICOM formatting [77] that contains information such as patient data. For this data the images have been scrubbed of identifying data and extracted as Portable Network Graphics (PNG) images. Axial CT scans done from the perspective of the axial or transverse plane, along or perpendicular to the median plane. In other words, with the patient lying on their back, slices are collected starting from the upper lobe of the lungs (closest to the patient's head) towards the lower lobe (closest to the patient's waist). Figure 5.1 shows an image sample for a single patient. Ranges and slice sizes vary for each patient. Therefore, I chose to focus on the mid section with the right major fissure in focus and the trachea splitting into the main bronchi (b) since this was fairly consistent among all patients 5.2.

Figure 5.2: From left to right, an image from a healthy patient, a patient with COVID-19, and a patient with non-COVID-19 illness

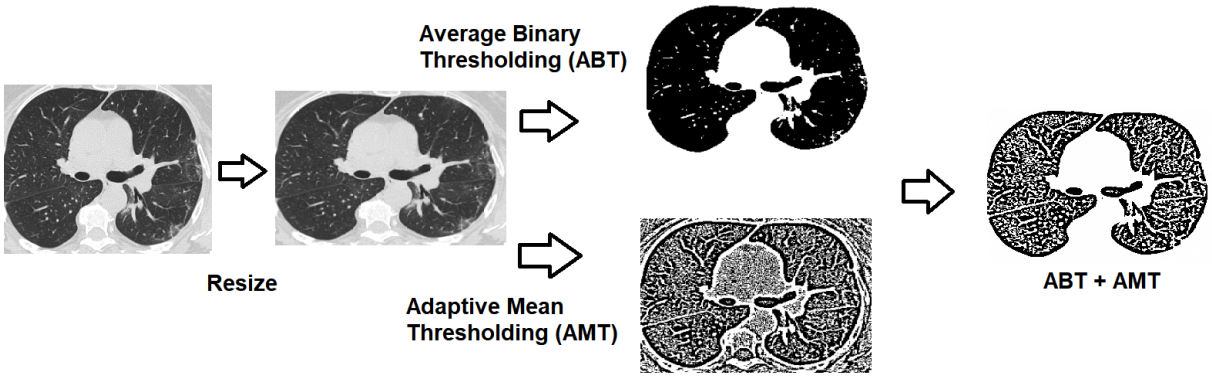

Figure 5.3: Example of the image transformation done to each CT scan in the data set. The image on the far left is the original image, which gets resized, then two separate thresholded images are extracted (ABT and AMT) which are added together to make up the final image that goes into the model

5.2.1 Encoding

For creating the HD vectors, I will use orthogonal mapping since it's been used the method of choice for encoding images [67, 93, 97]. Since this approach is dependent upon the position of the pixel, the images are resized to a predetermined size. For this project that size is 200 pixels high, and 300 pixels wide. I used OpenCV's [14] Python library for resizing.

Image thresholding

After resizing the images, they go through a thresholding filter. Using image thresholding is not unusual in machine learning-based image processing tasks[8, 63, 23], and commonly used for face [13] and object [29] detection.

The goal is to highlight the presence of fluid and debris inside the lung, therefore I use an average binary thresholding (ABT) filter, which is an absolute thresholding filter with the average pixel value of the image as the absolute threshold. Everything above the threshold will be mapped to 1 and everything below will be mapped to 0. This filter highlights artifacts and segments of the image that are not part of the lungs, such as the cardiac structure, bronchi, and surrounding tissue (see Figure 5.3). Separately, I apply an adaptive mean thresholding (AMT) filter which uses the algorithm described in [102] to dynamically calculate local thresholds according to a region size (in this case the region is 9 pixels, or a 3 by 3 pixel block). When adding the two filters, we are left with a highly contrasted image of the lungs with a reduced number of visual artifacts.

Normalizing the data

The final transformation is normalization by dividing all non-zero pixel values by 255. This makes the image binary (pixel values can only be 0 or 1).

Implementing orthogonal encoding

Once all the images are transformed, the encoding proceeds as follows (this is an adaptation of orthogonal encoding as described in [67]):

1. Set the dimensionality of the patient vector. In this case, 10k dimensional binary vectors.

2. Identify the magnitude of features. Since all the images are the same size after processing, then we have 300 by 200 pixels or 60,000 features.

3. Map each of the 60,000 pixels to a randomly generated vector. These vectors will be the same for all images (there are 60,000 unique vectors corresponding to the 60,000 pixels).

   If the value of the pixel is equal to 1, then the corresponding vector is added to the patient vector as is.

   If the value of the pixel is equal to 0, then the corresponding vector is transposed/shifted by 1 bit.

4. Add all the feature vectors using majority voting.

While simulating this algorithm, step 3's requirement for a unique vector for each pixel is achieved through NumPy's [83] (pseudo-)random number generator (RNG). The pixel index is the seed for the RNG used to generate the vectors. Therefore there's no need to keep an active vector library for pixel mapping.

5.2.2 Defining the population

From the 46 CT scans that didn't have any anomaly detected, 36 are randomly chosen to be used for training (leaving 10 for validation). That means, the encoded vectors of these 36 will be our known or training data. The prediction is based upon the Hamming distance from any of these vectors and the incoming ones. The remaining 10 vectors will become the control set, used for validating specificity. The 80 patients with COVID-19, and 67 with anomalies not related to COVID-19, are evaluated separately using the same training data.

5.2.3 Binary classification

A second model uses binary HD based classification (as described in [67]) between COVID-19 related anomalies and non-COVID-19 anomalies. This model was evaluated using leave-one-out cross validation [38]. Using the same encoded data from the HD-OCC model, classification proceeds as follows:

1. For each vector in the encoded data set, measure the hamming distance to all the remaining vectors (leave-one-out).

2. Predict the output class to that of the closest vector (with random tie-breaking).

   If the output class is the same as the known class of the vector being tested, then it is a true positive (if the class is COVID-19) or a true negative (if the class is not COVID-19).

   If the output classes don't match, then it's a false positive or false negative corresponding to whichever class was predicted.

This model serves to illustrate a scenario where specific anomalies (in this case COVID-19) are later discriminated from others (multi-class classification). A main advantage of HD

	COVID-19 anomalies	non-COVID-19 anomalies
AUC	0.72	0.74
Crossover	0.70	0.70
Specificity at Sensitivity		
0.90	0.20	0.20
0.70	0.70	0.70
0.50	0.80	0.90

Table 5.1: Area Under the Operating Curve (AUC), sensitivity and specificity crossover, and specificity values at sensitivity intervals 0.9, 0.7 and 0.5 for HD-OCC applied to CT scans from COVID-19 patients and non-COVID-19 patients separately.

computing is that we don't need to re encode the data in order to use it for other models. The binary classification described here uses the same encoded data as the HD-OCC model. With a bigger dataset we can explore the potential for cascading HD-OCC models that can detect multiple objects without relying on multi-class models.

5.3 Results

The models for COVID-19 anomalies and non-COVID-19 anomalies reported a 0.72 and 0.74 AUC respectively (see Table 5.1. The specificity and crossover values were very similar. The models can be easily adjusted to accommodate for higher specificity or sensitivity.

5.3.1 Binary classification

For the binary classification model with COVID-19 being the positive class, the model achieves 60% overall accuracy with 0.7 sensitivity and 0.5 specificity.

5.4 Comparing to expert radiologists

The performance of the binary classification model is comparable to the median sensitivity and specificity values presented in [7]. However, there are key differences between the two studies. In [7], CT scans from COVID-19 patients with no abnormalities were discarded. Additionally, radiologist had access to the full scan. For the model presented in this chapter, only one slice is being analyzed and all COVID-19 are included, without discarding non-anomalous images. Future research needs to be done to discover an encoding that is not dependent upon pixel position and that could be implemented to three-dimensional images.

5.5 Discussion

Compared to the approach used in chapter 3 and 4, image processing didn't require a lot of individual decision making on how to encode features. This is because in this case there's only one type of feature (a pixel) and the same encoding policy applies to all of them. The model performs well considering how difficult the task at hand is. From a more philosophical standpoint the model we built is the equivalent to having an expert radiologist identify if there's a anomaly on the CT scan using only his experience with CT scans form healthy patients.

There are some limitations to our models though. With state of the art deep learning algorithms there's very few manipulation of the image before going through the model. In chapter 3 I explain that HD computing's biggest advantage and disadvantage at the same time is how everything can and has to be heavily tailored for the problem. In the case of images, HD computing as it has been used for image processing, has little tolerance to context variations. The model expects the image to be at the same location with the same orientation every time. This is why we had to apply filters to maintain some uniformity

across the data. More research need to be done to discover a way to do automatic encoding and context detection with HD computing.

5.6 Summary

In this chapter I described HD-OCC applied to image encoding. I chose orthogonal encoding due to it being the encoding approach used in previous image classification tasks. Unlike the work described in Chapter 3 and Chapter 4, this time we have a *healthy* data set for training. To exemplify the use of one-class classification as a preliminary step for multi-class classification, I built a separate binary classification model that would use the anomalies detected from the HD-OCC model to separate COVID-19 anomalies and non-COVID-19 anomalies. Classification accuracy (60%) is comparable to the median accuracy from expert radiologists, considering some key differences. This project shows that HD-OCC shares the same versatility of general HD Computing classification.

Chapter 6

Related work

While HD computing has been around for a little more than a decade [60], the core architectural ideas behind it were developed in the early 1990s [59]. Likewise, machine learning, and more specifically OCC, has been used to solve health informatics' problems before. In this chapter I give a brief overview of the most relevant work in these areas that is closely related, and some served as inspiration, to this dissertation's research.

6.1 Neuro-inspired architectures

HD computing is not the only effort to build neuro-inspired architectures. There's broad research on different approaches to building frameworks and hardware that perform cognitive computing straightforward with minimal software simulation. Some of the most important work in this field has been done under the flag of neuromorphic architectures [74]. The aim of this type of architectures it to emulate neural activity and neuron morphology through hardware implementation. Neuromorphic architectures commonly used analog circuits but it has evolved into several approaches that include digital and hybrid systems [80]. IBM

has been one of the major sponsors of neuromorphic computing achieving neural computing performance comparable to supercomputers at a fraction of the cost [49].

Coming back to HD computing, even though it was proposed more than a decade ago, it is only in recent years that application oriented research started to become public. To my knowledge, there is no other project that has dealt with using HD computing to process patient data.

One of the most prominent works in HD computing for bioinformatics, which is closely related to health informatics, is the work of Rahimi, et al. [91, 90]. data from non-invasive electrode's is used to model brain activity and predict the subject's intentions. They achieve a 5% improvement in accuracy over machine learning approaches. Similarly, Imani, et al. [52] uses HD computing for DNA modeling, achieving over 99% accuracy.

In conjunction to HD computing applied to bioinformatics, Imani, et al. [53] proposes an FPGA architecture that achieves over 5X performance when compared to other FPGA approaches. This is done by emulating an associative memory module with custom hardware configuration. Furthermore, Salamat, et al. [98] describes an architecture that is up to 11 times more energy efficient than a GPU implementation. These findings are key factors for justifying the use of HD Computing in constrained environments such as, in the case of sepsis modeling, next to a patient's bed.

6.2 Machine learning and health informatics

The domain of health informatics is broad and full of challenges. Machine learning has been used to improve patient care, mine qualitative information, enhance preventative care, analyze hospital statistics, among others[48]. Even in the middle of a fast evolving pandemic, the majority of published research related to COVID-19 is centered on artificial intelligence

and machine learning [111, 55]. Some of the work is centered on using CT scans as input [1, 94, 2, 108] but this is still an evolving problem and most results are experimental [6].

In the case of diabetes, most machine learning research is focused on predicting before the patient is diagnosed with the condition. With the greater availability of data and the accuracy that deep learning achieves, some of the research has seen real life implementation [62, 3, 26]. In the specific case of the Pima Indian dataset, most recent research achieves up to 98% classification accuracy using deep learning [81]. It is important to note, however, that due to the size of the dataset, validation accuracy is unlikely to translate into implementation accuracy. There's promising research with bigger datasets that is being translated into real world applications [27]. The most resilient ones are self-improving and self-sustainable by feeding from the data they process. This is what allows novel approaches achieve the development study phase of research and eventually implementation [100].

Similarly, sepsis has also been targeted by ML research. The key difference between diabetes and sepsis is that the later is time critical. While diabetes can take years to develop and even longer to become fatal, sepsis evolves in a matter of hours, therefore solutions need to take real time performance into account. In chapter 4 I describe some of the work that is closely related to the problem used for HD-OCC, but in addition to it, it is crucial to refer to the work of Ginestra, et al. [42]. The focus of this work is to evaluate the perception of the medical staff towards ML based alarm system for septic shock called Early Warning System 2.0. They discovered that only about 40% of the surveyed nurses had a favorable opinion and found the system useful. When surveying primary care providers, their confidence in the system drops to 16%. The mayor issues raised by those who didn't trust the system were that they didn't understand it, they trusted their instincts more, or they found the system to be redundant. The work of Nemati, et. al. [82] circumvents some of these issues by focusing on explainable machine learning for sepsis, but there's plenty of work to be done to understand what nurses and doctors need or are likely to accept and have medical driven

research lead the development of new systems.

6.3 OCC in health informatics

As mentioned in chapter 2, research in OCC is scarce. I suspect that a key factor for this is that most OCC approaches involve modifying binary or multi-class models to account for imbalanced datasets. This is likely to produce sub-optimal results. However, there's previous research on using OCC for medical image filtering and image classification[39, 32].

To my knowledge, the work of Mourão-Miranda, et al. [78] is the first to suggest using OCC for patient classification. The use case implements SVM based OCC to identify patients with depression (with depressed being the anomaly class). Similarly, Gomes, et al. [16] uses OCC to identify different types of heart diseases. Their baseline is established using SVM-OCC but the main contribution is the use of *Features Boundaries Detection for One-class Classification*, an approach that builds a hypersphere using features as dimensional parameters to determine membership to the training class.

6.4 Synthetic patient datasets

Synthetic data has found a renewed application in health informatics. While being a relatively old concept, the need for synthetic data fades away as more real data is made available. However, in health informatics data availability is still a major issue. Most of the research in this area is centered on generating high quality synthetic data for deep learning tasks. In the work of Choi, et al. [21] synthetic patient records are generated using adversarial networks. An approach echoed by Guibas, et al. [45] but in this case applied towards medical images. Data scarcity is a bigger problem when specific sequences are needed, like the ones used by

long short-term memory (LSTM) networks. This is why Baytas, et al. [10] uses synthetic data to validate and train their LSTM model for patient clustering.

In this dissertation I decided to use data generation approaches that were not performance intensive. As more data is made available I can change the approach. There's a need for a detailed review on how synthetic data impacts HD performance, and determine if there's an approach that is best suited for this paradigm.

6.5 Discussion

Most of the approaches used in this dissertation research (synthetic dataset, OCC, HD computing) have extensive preliminary work. Until now, they hadn't been combined to study how they work with each other. While there is much work to be done in discovering the full potential of HD-OCC, research from other domains can guide us to find the answers we are looking for.

Towards the end of the paper, Kanerva (2009) delves into philosophical questions on the possibility of creating artificial brains that work just as the target brains they are trying to emulate. In this section he writes: "If our theories allow us to build a system whose behavior is indistinguishable from the behavior of the intended *target* system, we have understood that system—the theory embodies our understanding of it. This view places the burden on modeling" (p157). Staying with this thought, and continuing the argument set forth in chapter 1, our current understanding on how human brains do recognition seems to point towards an OCC-like behavior. HD-OCC is, according to this research, the closest we have to modeling classification as a cognitive skill. If this statement is proven accurate, then we are one step closer to understanding the *target* system. If it's not accurate then we understand that this is not how the system works.

Chapter 7

Conclusion

In this dissertation I described HD-OCC, a one-class classification (OCC) implementation using hyperdimensional (HD) computing. This approach was tested using three key domains: linear features, sequential features, and image processing. In general, HD computing is ideal for OCC because it uses a simple approach to classification. Relying on a distance measure, boundaries are not entirely dependent upon differences between classes but can be controlled with distance thresholds. This is a major advantage over artificial neural networks and other ML approaches that can't directly identify non-membership learning from a single class. Additionally, HD-OCC outperforms classical ML approaches to OCC.

7.1 Main Contributions

The main contributions fo this dissertation are as follows:

- A general framework for applying HD-OCC to different problem domains.
- A formula based approach for dynamic generation of encoding vectors for linear en-

coding.

- A programming library-aided approach to dynamic generation of HD vectors for orthogonal encoding.
- First time implementation of synthetic dataset for HD computing.
- First time combining sequential and linear features in the same hypervector.
- Introduced *linear n-gram* encoding, a sequential encoding method that preserves linearity.
- A general decision making guideline for proper feature encoding.

7.2 Future Work

Further evaluation of the impact of synthetic datasets in HD computing is required to establish a framework for proper generation. Furthermore, each of the implementations presented in this dissertation would benefit from additional access to data. Eventually HD-OCC should become part of a larger HD computing model that performs hierarchical classification in a way that is inspired by cognitive processes in the human brain.

Finally, HD-OCC would benefit from real-life implementation. Especially when dealing with patient data. The model presented in chapter 4 for sepsis, for example, can be implemented as a bedside solution for patient monitoring. Allowing for such an implementation to become true would help us better understand the role that HD computing has in the future of artificial intelligence.

Bibliography

[1] P. Afshar, S. Heidarian, N. Enshaei, F. Naderkhani, M. J. Rafiee, A. Oikonomou, F. B. Fard, K. Samimi, K. N. Plataniotis, and A. Mohammadi. Covid-ct-md: Covid-19 computed tomography (ct) scan dataset applicable in machine learning and deep learning. *arXiv preprint arXiv:2009.14623*, 2020.

[2] D. Al-Karawi, S. Al-Zaidi, N. Polus, and S. Jassim. Machine learning analysis of chest ct scan images as a complementary digital test of coronavirus (covid-19) patients. *medRxiv*, 2020.

[3] B. Alić, L. Gurbeta, and A. Badnjević. Machine learning techniques for classification of diabetes and cardiovascular diseases. In *2017 6th Mediterranean Conference on Embedded Computing (MECO)*, pages 1–4. IEEE, 2017.

[4] J. R. Anderson. *Cognitive skills and their acquisition*. Psychology Press, 1981.

[5] T. Avni, A. Lador, S. Lev, L. Leibovici, M. Paul, and A. Grossman. Vasopressors for the treatment of septic shock: systematic review and meta-analysis. *PloS one*, 10(8):e0129305, 2015.

[6] P. Bachtiger, N. S. Peters, and S. L. Walsh. Machine learning for covid-19—asking the right questions. *The Lancet Digital Health*, 2(8):e391–e392, 2020.

[7] H. X. Bai, B. Hsieh, Z. Xiong, K. Halsey, J. W. Choi, T. M. L. Tran, I. Pan, L.-B. Shi, D.-C. Wang, J. Mei, et al. Performance of radiologists in differentiating covid-19 from viral pneumonia on chest ct. *Radiology*, page 200823, 2020.

[8] T. Bangira, S. M. Alfieri, M. Menenti, and A. Van Niekerk. Comparing thresholding with machine learning classifiers for mapping complex water. *Remote Sensing*, 11(11):1351, 2019.

[9] P. Barrouillet. Theories of cognitive development: From piaget to today, 2015.

[10] I. M. Baytas, C. Xiao, X. Zhang, F. Wang, A. K. Jain, and J. Zhou. Patient subtyping via time-aware lstm networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 65–74, 2017.

[11] D. Berlinski. *The advent of the algorithm: the 300-year journey from an idea to the computer*. Houghton Mifflin Harcourt, 2001.

[12] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and information systems*, 34(3):483–519, 2013.

[13] D. Bradley and G. Roth. Adaptive thresholding using the integral image. *Journal of graphics tools*, 12(2):13–21, 2007.

[14] G. Bradski and A. Kaehler. Opencv. *Dr. Dobb's journal of software tools*, 3, 2000.

[15] J. S. Bruner and G. A. Austin. *A study of thinking*. Transaction publishers, 1986.

[16] G. G. Cabral and A. L. I. de Oliveira. One-class classification for heart disease diagnosis. In *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2551–2556. IEEE, 2014.

[17] F. Candel, E. Grima, M. Matesanz, C. Cervera, G. Soto, M. Almela, J. Martınez, M. Navasa, F. Cofan, M. Ricart, et al. Bacteremia and septic shock after solid-organ transplantation. In *Transplantation proceedings*, volume 37, pages 4097–4099. Elsevier, 2005.

[18] A. Carfì, R. Bernabei, F. Landi, et al. Persistent symptoms in patients after acute covid-19. *Jama*, 324(6):603–605, 2020.

[19] R. Castro, P. M. Luz, M. D. Wakimoto, V. G. Veloso, B. Grinsztejn, and H. Perazzo. Covid-19: a meta-analysis of diagnostic test accuracy of commercial assays registered in brazil. *The Brazilian Journal of Infectious Diseases*, 2020.

[20] Y. Chen, X. S. Zhou, and T. S. Huang. One-class svm for learning in image retrieval. In *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*, volume 1, pages 34–37. IEEE, 2001.

[21] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun. Generating multi-label discrete patient records using generative adversarial networks. *arXiv preprint arXiv:1703.06490*, 2017.

[22] P. A. Chou. The capacity of the kanerva associative memory. *IEEE Transactions on Information Theory*, 35(2):281–298, 1989.

[23] A. Chowdhury, E. Kautz, B. Yener, and D. Lewis. Image driven machine learning methods for microstructure recognition. *Computational Materials Science*, 123:176–187, 2016.

[24] E. Coiera. *Guide to health informatics*. CRC press, 2015.

[25] E. R. F. Collaboration et al. Diabetes mellitus, fasting blood glucose concentration, and risk of vascular disease: a collaborative meta-analysis of 102 prospective studies. *The Lancet*, 375(9733):2215–2222, 2010.

[26] A. Dagliati, S. Marini, L. Sacchi, G. Cogni, M. Teliti, V. Tibollo, P. De Cata, L. Chiovato, and R. Bellazzi. Machine learning methods to predict diabetes complications. *Journal of diabetes science and technology*, 12(2):295–302, 2018.

[27] I. Dankwa-Mullan, M. Rivo, M. Sepulveda, Y. Park, J. Snowdon, and K. Rhee. Transforming diabetes care through artificial intelligence: the future is here. *Population health management*, 22(3):229–242, 2019.

[28] M. Davies and P.-O. Hagen. Systemic inflammatory response syndrome. *British Journal of Surgery*, 84(7):920–935, 1997.

[29] M. P. De Albuquerque, I. A. Esquef, and A. G. Mello. Image thresholding using tsallis entropy. *Pattern Recognition Letters*, 25(9):1059–1065, 2004.

[30] R. P. Dellinger, M. M. Levy, A. Rhodes, D. Annane, H. Gerlach, S. M. Opal, J. E. Sevransky, C. L. Sprung, I. S. Douglas, R. Jaeschke, et al. Surviving sepsis campaign: international guidelines for management of severe sepsis and septic shock, 2012. *Intensive care medicine*, 39(2):165–228, 2013.

[31] J. G. Derraik, M. Rademaker, W. S. Cutfield, T. E. Pinto, S. Tregurtha, A. Faherty, J. M. Peart, P. L. Drury, and P. L. Hofman. Effects of age, gender, bmi, and anatomical site on skin thickness in children and adults with diabetes. *PLoS One*, 9(1):e86637, 2014.

[32] C. Désir, S. Bernard, C. Petitjean, and L. Heutte. A random forest based approach for one class classification in medical imaging. In *International Workshop on Machine Learning in Medical Imaging*, pages 250–257. Springer, 2012.

[33] J. J. DiCarlo, D. Zoccolan, and N. C. Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012.

[34] S. Dugar, C. Choudhary, and A. Duggal. Sepsis and septic shock: Guideline-based management. *Cleveland Clinic Journal of Medicine*, 87(1):53–64, 2020.

[35] X. Fang, Z. Wang, J. Yang, H. Cai, Z. Yao, K. Li, and Q. Fang. Clinical evaluation of sepsis-1 and sepsis-3 in the icu. *Chest*, 153(5):1169–1176, 2018.

[36] D. S. Feig and V. A. Palda. Type 2 diabetes in pregnancy: a growing concern. *The Lancet*, 359(9318):1690–1692, 2002.

[37] A. R. Frisancho. Triceps skin fold and upper arm muscle size norms for assessment of nutritional status. *The American journal of clinical nutrition*, 27(10):1052–1058, 1974.

[38] K. Fukunaga and D. M. Hummels. Leave-one-out procedures for nonparametric error estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(4):421–423, 1989.

[39] L. Gao, L. Zhang, C. Liu, and S. Wu. Handling imbalanced medical image data: A deep-learning-based one-class classification approach. *Artificial Intelligence in Medicine*, 108:101935, 2020.

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

[41] H. M. Giannini, J. C. Ginestra, C. Chivers, M. Draugelis, A. Hanish, W. D. Schweickert, B. D. Fuchs, L. Meadows, M. Lynch, P. J. Donnelly, et al. A machine learning algorithm to predict severe sepsis and septic shock: development, implementation, and impact on clinical practice. *Read Online: Critical Care Medicine— Society of Critical Care Medicine*, 47(11):1485–1492, 2019.

[42] J. C. Ginestra, H. M. Giannini, W. D. Schweickert, L. Meadows, M. J. Lynch, K. Pavan, C. J. Chivers, M. Draugelis, P. J. Donnelly, B. D. Fuchs, et al. Clinician perception of a machine learning–based early warning system designed to predict severe sepsis and septic shock. *Critical care medicine*, 47(11):1477–1484, 2019.

[43] H. H. Goldstine. *The computer from Pascal to von Neumann*. Princeton University Press, 1993.

[44] A. Goncalves, P. Ray, B. Soper, J. Stevens, L. Coyle, and A. P. Sales. Generation and evaluation of synthetic patient data. *BMC Medical Research Methodology*, 20:1–40, 2020.

[45] J. T. Guibas, T. S. Virdi, and P. S. Li. Synthetic medical images from dual generative adversarial networks. *arXiv preprint arXiv:1709.01872*, 2017.

[46] H. He, Y. Bai, E. A. Garcia, and S. Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. IEEE, 2008.

[47] T. A. Hillier and K. L. Pedula. Characteristics of an adult population with newly diagnosed type 2 diabetes: the relation of obesity and age of onset. *Diabetes care*, 24(9):1522–1527, 2001.

[48] A. Holzinger. Machine learning for health informatics. In *Machine Learning for Health Informatics*, pages 1–24. Springer, 2016.

[49] J. Hsu. Ibm's new brain [news]. *IEEE spectrum*, 51(10):17–19, 2014.

[50] P. Huang, T. Liu, L. Huang, H. Liu, M. Lei, W. Xu, X. Hu, J. Chen, and B. Liu. Use of chest ct in combination with negative rt-pcr assay for the 2019 novel coronavirus but high clinical suspicion. *Radiology*, 295(1):22–23, 2020.

[51] M. Imani, D. Kong, A. Rahimi, and T. Rosing. Voicehd: Hyperdimensional computing for efficient speech recognition. In *2017 IEEE International Conference on Rebooting Computing (ICRC)*, pages 1–8. IEEE, 2017.

[52] M. Imani, T. Nassar, A. Rahimi, and T. Rosing. Hdna: Energy-efficient dna sequencing using hyperdimensional computing. In *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, pages 271–274. IEEE, 2018.

[53] M. Imani, S. Salamat, S. Gupta, J. Huang, and T. Rosing. Fach: Fpga-based acceleration of hyperdimensional computing by reducing computational complexity. In *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, pages 493–498, 2019.

[54] B. Inhelder and J. Piaget. La genèse des structures logiques élémentaires classifications et sériations. 1959.

[55] M. Jamshidi, A. Lalbakhsh, J. Talla, Z. Peroutka, F. Hadjilooei, P. Lalbakhsh, M. Jamshidi, L. La Spada, M. Mirmozafari, M. Dehghani, et al. Artificial intelligence and covid-19: deep learning approaches for diagnosis and treatment. *IEEE Access*, 8:109581–109595, 2020.

[56] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-Wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[57] A. Joshi, J. T. Halseth, and P. Kanerva. Language geometry using random indexing. In *International Symposium on Quantum Interaction*, pages 265–274. Springer, 2016.

[58] P. Kanerva. *Sparse distributed memory*. MIT press, 1988.

[59] P. Kanerva. Sparse distributed memory and related models. 1992.

[60] P. Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive computation*, 1(2):139–159, 2009.

[61] S. Kasen, P. Cohen, H. Chen, and A. Must. Obesity and psychopathology in women: a three decade prospective study. *International Journal of Obesity*, 32(3):558–566, 2008.

[62] I. Kavakiotis, O. Tsave, A. Salifoglou, N. Maglaveras, I. Vlahavas, and I. Chouvarda. Machine learning and data mining methods in diabetes research. *Computational and structural biotechnology journal*, 15:104–116, 2017.

[63] J. Ker, S. P. Singh, Y. Bai, J. Rao, T. Lim, and L. Wang. Image thresholding improves 3-dimensional convolutional neural network diagnosis of different acute brain hemorrhages on computed tomography scans. *Sensors*, 19(9):2167, 2019.

[64] S. S. Khan and M. G. Madden. A survey of recent trends in one class classification. In *Irish conference on artificial intelligence and cognitive science*, pages 188–197. Springer, 2009.

[65] K. M. Kiely. *Cognitive Function*, pages 974–978. Springer Netherlands, Dordrecht, 2014.

[66] J. Kim, H. Chang, D. Kim, D.-H. Jang, I. Park, and K. Kim. Machine learning for prediction of septic shock at initial triage in emergency department. *Journal of critical care*, 55:163–170, 2020.

[67] D. Kleyko, A. Rahimi, D. A. Rachkovskij, E. Osipov, and J. M. Rabaey. Classification and recall with binary hyperdimensional computing: Tradeoffs in choice of density and mapping characteristics. *IEEE transactions on neural networks and learning systems*, 29(12):5880–5898, 2018.

[68] W. C. Knowler, D. J. Pettitt, P. J. Savage, and P. H. Bennett. Diabetes incidence in pima indians: contributions of obesity and parental diabetes. *American journal of epidemiology*, 113(2):144–156, 1981.

[69] B. Levy, S. Collin, N. Sennoun, N. Ducrocq, A. Kimmoun, P. Asfar, P. Perez, and F. Meziani. Vascular hyporesponsiveness to vasopressors in septic shock: from bench to bedside. In *Applied Physiology in Intensive Care Medicine 2*, pages 251–261. Springer, 2012.

[70] L. Li, L. Qin, Z. Xu, Y. Yin, X. Wang, B. Kong, J. Bai, Y. Lu, Z. Fang, Q. Song, et al. Artificial intelligence distinguishes covid-19 from community acquired pneumonia on chest ct. *Radiology*, 2020.

[71] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[72] M. Luo, K. Wang, Z. Cai, A. Liu, Y. Li, and C. F. Cheang. Using imbalanced triangle synthetic data for machine learning anomaly detection. *Comput., Mater. Continua*, 58(1):15–26, 2019.

[73] P. McCorduck and C. Cfe. *Machines who think: A personal inquiry into the history and prospects of artificial intelligence*. CRC Press, 2004.

[74] C. Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636, 1990.

[75] A. Melamed and F. J. Sorvillo. The burden of sepsis-associated mortality in the united states from 1999 to 2005: an analysis of multiple-cause-of-death data. *Critical care*, 13(1):R28, 2009.

[76] D. Mellitus. Diagnosis and classification of diabetes mellitus. *Diabetes care*, 28(S37):S5–S10, 2005.

[77] P. Mildenberger, M. Eichelberg, and E. Martin. Introduction to the dicom standard. *European radiology*, 12(4):920–927, 2002.

[78] J. Mourão-Miranda, D. R. Hardoon, T. Hahn, A. F. Marquand, S. C. Williams, J. Shawe-Taylor, and M. Brammer. Patient classification as an outlier detection problem: an application of the one-class support vector machine. *Neuroimage*, 58(3):793–804, 2011.

[79] M. M. Moya and D. R. Hush. Network constraints and multi-objective optimization for one-class classification. *Neural networks*, 9(3):463–474, 1996.

[80] R. A. Nawrocki, R. M. Voyles, and S. E. Shaheen. A mini review of neuromorphic architectures and implementations. *IEEE Transactions on Electron Devices*, 63(10):3819–3829, 2016.

[81] H. Naz and S. Ahuja. Deep learning approach for diabetes prediction using pima indian dataset. *Journal of Diabetes & Metabolic Disorders*, 19(1):391–403, 2020.

[82] S. Nemati, A. Holder, F. Razmi, M. D. Stanley, G. D. Clifford, and T. G. Buchman. An interpretable machine learning model for accurate prediction of sepsis in the icu. *Critical care medicine*, 46(4):547, 2018.

[83] T. E. Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[84] G. Palm. On associative memory. *Biological cybernetics*, 36(1):19–31, 1980.

[85] F. Pan, T. Ye, P. Sun, S. Gui, B. Liang, L. Li, D. Zheng, J. Wang, R. L. Hesketh, L. Yang, et al. Time course of lung changes on chest ct during recovery from 2019 novel coronavirus (covid-19) pneumonia. *Radiology*, 2020.

[86] S. Pass. *Parallel paths to constructivism: Jean Piaget and Lev Vygotsky*. IAP, 2004.

[87] P. Perera and V. M. Patel. Learning deep features for one-class classification. *IEEE Transactions on Image Processing*, 28(11):5450–5463, 2019.

[88] D. H. Perkel and T. H. Bullock. Neural coding. *Neurosciences Research Program Bulletin*, 1968.

[89] T. J. Pollard, A. E. Johnson, J. D. Raffa, L. A. Celi, R. G. Mark, and O. Badawi. The eicu collaborative research database, a freely available multi-center database for critical care research. *Scientific data*, 5:180178, 2018.

[90] A. Rahimi, P. Kanerva, L. Benini, and J. M. Rabaey. Efficient biosignal processing using hyperdimensional computing: Network templates for combined learning and classification of exg signals. *Proceedings of the IEEE*, 107(1):123–143, 2018.

[91] A. Rahimi, P. Kanerva, J. d. R. Millán, and J. M. Rabaey. Hyperdimensional computing for noninvasive brain-computer interfaces: Blind and one-shot classification of eeg error-related potentials. In *10th EAI Int. Conf. on Bio-inspired Information and Communications Technologies*, number CONF, 2017.

[92] D. Ravì, C. Wong, F. Deligianni, M. Berthelot, J. Andreu-Perez, B. Lo, and G.-Z. Yang. Deep learning for health informatics. *IEEE journal of biomedical and health informatics*, 21(1):4–21, 2016.

[93] G. Recchia, M. Sahlgren, P. Kanerva, and M. N. Jones. Encoding sequential information in semantic space models: Comparing holographic reduced representation and random permutation. *Computational intelligence and neuroscience*, 2015, 2015.

[94] M. Roberts, D. Driggs, M. Thorpe, J. Gilbey, M. Yeung, S. Ursprung, A. I. Aviles-Rivero, C. Etmann, C. McCague, L. Beer, et al. Machine learning for covid-19 detection and prognostication using chest radiographs and ct scans: a systematic methodological review. *arXiv preprint arXiv:2008.06388*, 2020.

[95] B. R. Rohrer. Biologically inspired feature creation for multi-sensory perception. Technical report, Sandia National Lab.(SNL-NM), Albuquerque, NM (United States), 2011.

[96] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402, 2018.

[97] M. Sahlgren, A. Holst, and P. Kanerva. Permutations as a means to encode order in word space. In *The 30th Annual Meeting of the Cognitive Science Society (CogSci'08), 23-26 July 2008, Washington DC, USA*, 2008.

[98] S. Salamat, M. Imani, B. Khaleghi, and T. Rosing. F5-hd: Fast flexible fpga-based framework for refreshing hyperdimensional computing. In *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 53–62, 2019.

[99] C. W. Seymour, V. X. Liu, T. J. Iwashyna, F. M. Brunkhorst, T. D. Rea, A. Scherag, G. Rubenfeld, J. M. Kahn, M. Shankar-Hari, M. Singer, et al. Assessment of clinical criteria for sepsis: for the third international consensus definitions for sepsis and septic shock (sepsis-3). *Jama*, 315(8):762–774, 2016.

[100] J. Shen, J. Chen, Z. Zheng, J. Zheng, Z. Liu, J. Song, S. Y. Wong, X. Wang, M. Huang, P.-H. Fang, et al. An innovative artificial intelligence–based app for the diagnosis of gestational diabetes mellitus (gdm-ai): Development study. *Journal of Medical Internet Research*, 22(9):e21573, 2020.

[101] M. Singer, C. S. Deutschman, C. W. Seymour, M. Shankar-Hari, D. Annane, M. Bauer, R. Bellomo, G. R. Bernard, J.-D. Chiche, C. M. Coopersmith, et al. The third international consensus definitions for sepsis and septic shock (sepsis-3). *Jama*, 315(8):801–810, 2016.

[102] T. R. Singh, S. Roy, O. I. Singh, T. Sinam, K. Singh, et al. A new local adaptive thresholding technique in binarization. *arXiv preprint arXiv:1201.5227*, 2012.

[103] J. W. Smith, J. Everhart, W. Dickson, W. Knowler, and R. Johannes. Using the adap learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Annual Symposium on Computer Application in Medical Care*, page 261. American Medical Informatics Association, 1988.

[104] M. G. Smith and L. Bull. Using genetic programming for feature creation with a genetic algorithm feature selector. In *International Conference on Parallel Problem Solving from Nature*, pages 1163–1171. Springer, 2004.

[105] E. Soares, P. Angelov, S. Biaso, M. H. Froes, and D. K. Abe. Sars-cov-2 ct-scan dataset: A large dataset of real patients ct scans for sars-cov-2 identification. *medRxiv*, 2020.

[106] C. P. Subbe, M. Kruger, P. Rutherford, and L. Gemmel. Validation of a modified early warning score in medical admissions. *Qjm*, 94(10):521–526, 2001.

[107] D. M. J. Tax. One-class classification: Concept learning in the absence of counter-examples. 2002.

[108] A. Ter-Sarkisov. Covid-ct-mask-net: Prediction of covid-19 from ct scans using regional features. *medRxiv*, 2020.

[109] I. B. A. TURING. Computing machinery and intelligence-am turing. *Mind*, 59(236):433, 1950.

[110] S. Tusgul, P.-N. Carron, B. Yersin, T. Calandra, and F. Dami. Low sensitivity of qsofa, sirs criteria and sepsis definition to identify infected patients at risk of complication in the prehospital setting and at the emergency department triage. *Scandinavian journal of trauma, resuscitation and emergency medicine*, 25(1):1–7, 2017.

[111] R. Vaishya, M. Javaid, I. H. Khan, and A. Haleem. Artificial intelligence (ai) applications for covid-19 pandemic. *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, 2020.

[112] J.-L. Vincent, R. Moreno, J. Takala, S. Willatts, A. De Mendonça, H. Bruining, C. Reinhart, P. Suter, and L. G. Thijs. The sofa (sepsis-related organ failure assessment) score to describe organ dysfunction/failure, 1996.

[113] J. Von Neumann. First draft of a report on the edvac. *IEEE Annals of the History of Computing*, 15(4):27–75, 1993.

[114] L. Vygotsky. Consciousness as a problem in the psychology of behavior. *Soviet psychology*, 17(4):3–35, 1979.

[115] H. M. Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984, 2006.

[116] B. Xu, Y. Xing, J. Peng, Z. Zheng, W. Tang, Y. Sun, C. Xu, and F. Peng. Chest ct for detecting covid-19: a systematic review and meta-analysis of diagnostic accuracy. *European Radiology*, page 1, 2020.

[117] N. Zhu, D. Zhang, W. Wang, X. Li, B. Yang, J. Song, X. Zhao, B. Huang, W. Shi, R. Lu, et al. A novel coronavirus from patients with pneumonia in china, 2019. *New England Journal of Medicine*, 2020.