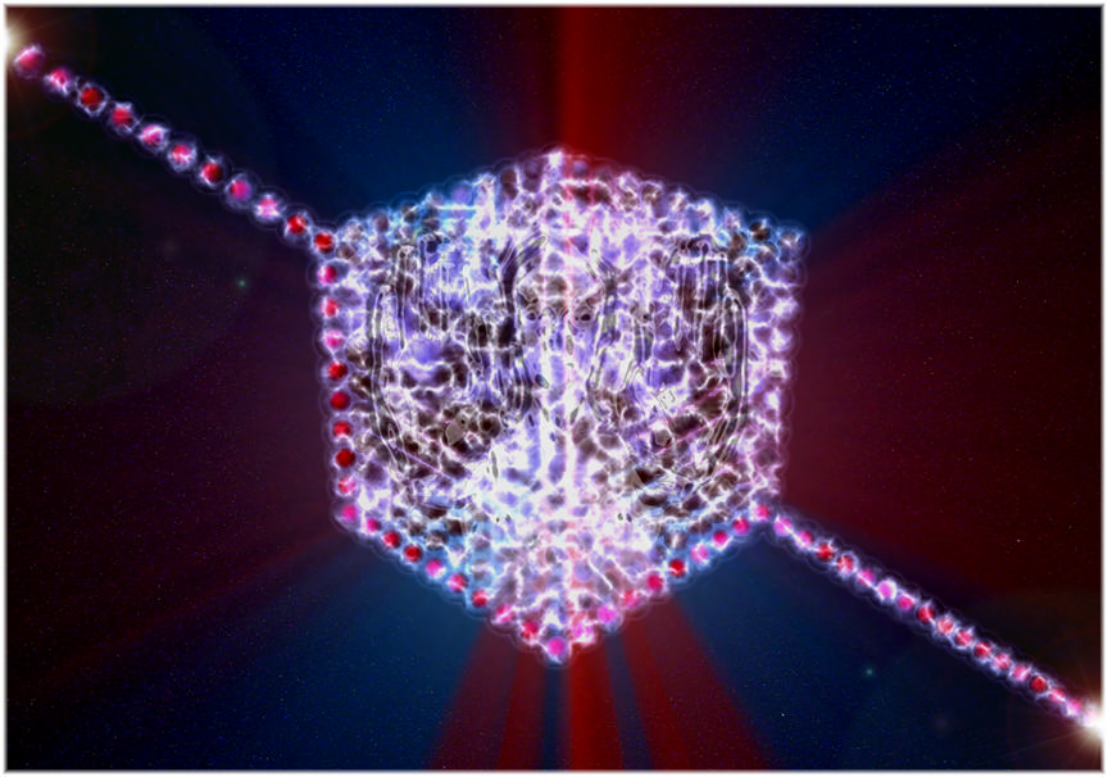
MASTER'S THESIS

Henrik Daniel Kjeldsen - s1635506 - H.D.Kjeldsen@student.rug.nl
Taco Mesdagstraat 13A, 9718KH Groningen, The Netherlands, 2008

# Automatic Vowel Recognition with GPU based Holographic Neural Network



**Supervisor: Dr. Ronald van Elburg**
**2nd Supervisor: Dr. Tjeerd Andringa**
**Auditory Cognition Group**

university of groningen / faculty of mathematics and natural sciences / artificial intelligence

# Abstract

Distributed representation in the brain implies that neural representations are patterns of activity over many neurons and that the same neurons participate in many patterns.

Holographic neural networks achieve distributed representation through a mathematical analogy to optical holography. Besides being aesthetically pleasing, the holographic analogy promises highly effective search and inference capabilities and allows robust representations that degrade gracefully.

We build upon existing holographic neural networks and implement an important improvement to the paradigm that removes a previous restriction to feature vectors that are of random distribution. This means that it is no longer necessary to map between natural (not random) signal features and a set of random features; instead the signal features can be used directly.

We develop a simple holographic neural network classifier and apply it to the AI-task of automatic vowel recognition with good results that demonstrate the feasibility of the improvement.

The system features a very simple learning scheme adapted from earlier holographic neural networks and uses a parallel graphics processing unit to accelerate both learning and classification.

We also give suggestions for further research on holographic neural networks aimed at more difficult AI-tasks, like automatic speech recognition.

**Table of Contents**

# 1. Introduction

The main objective of this thesis is to demonstrate the application of a novel holographic neural network (HNN) to the entry-level AI task of automatic vowel recognition (AVR).
By an *entry-level* task we mean a task that by design has been limited in its cognitive requirements so that difficult issues like general world-knowledge can be disregarded.
We believe that successful demonstration on an entry-level AI task is the first step towards more sophisticated HNN that can eventually take on more difficult AI tasks.
We consider AVR to be an entry-level AI task at the beginning of a development path to automatic speech recognition (ASR). The full ASR task is assumed to be too difficult for an AI system that does not consider complex high-level cognitive processes like intuition, emotion and empathy, and does not possess general world-knowledge; e.g. a deep semantic understanding is assumed to be necessary for human-level recognition of sentence level series of spoken words in realistic environments. We do not consider this kind of semantic context on the AVR task, instead we look at the simple similarity of low-level data patterns; this is also needed for ASR, but alone it is not enough.
Although we do not model high-level brain processes we still believe that the brain is an essential reference system, and in extension that AI (aiming at human-level performance) must exhibit a high degree of plausibility in relation to the brain, but not just any plausibility (e.g. an AI system in a skull does not mean that it is plausible in relation to the brain just because the brain is also in a skull), we must therefore try to establish key principles of cognition in the brain as plausibility criteria; in the next section we call this a reverse-engineering perspective.
We assume that in such brain inspired AI, in order to achieve plausibility, we must aim to connect low-level neural processes with high-level cognitive processes. Approaching this task from the top might feel like the most natural strategy considering we all have introspective access to some high-level cognitive processes, but it has proven difficult to produce AI systems that model high-level processes and at the same time allow the level of analysis to be moved all the way down to realistic low-level neural representations.
HNN takes the opposite bottom-up approach starting with low-level distributed representations that are assumed to be neurally plausible (in the sense that they are distributed over many neurons), and then build up to high-level cognitive constructs. On the AVR task a minimum of high-level processing is needed, so AVR is a natural entry-point for HNN.

The system we develop also considers a secondary point of brain plausibility; i.e. parallel processing in the sense that specific computationally expensive sections are accelerated by a parallel graphics processing unit (GPU). In the next section we will see that we do not consider this to be a strong kind of plausibility, because we do not believe parallel processing to be a key principle of cognition in other terms than computational speed.

The following pages are structured as follows:
The subsequent section is an essay trying to establish key principles of cognition and motivating the choice of HNN as an AI approach with plausible low-level distributed representations. This is followed by a more in depth treatment of the theoretical background of HNN leading up to the development of a potential improvement to the paradigm, and of a simple HNN classifier incorporating the improvement. We then review the AVR task and typical datasets and results, and finally apply the novel HNN classifier to AVR and present and discuss the results obtained.
Along the way we draw further perspectives to ASR and describe possible future directions for HNN for this more difficult kind of AI task.

# 2. The Quest for AI

This section is an introductory essay taking a reverse-engineering perspective on the scientific quest for AI. Its purpose is to set the stage for the sections ahead, which concern the development of a specific AI system guided by the principles established here.

Let us start by briefly discussing a definition of AI: An agreed and universally accepted definition of AI is very difficult to achieve, not least, because a consistent definition of intelligence itself is hard to pin down, and there are even different opinions on which levels of intelligence AI should aim for (see e.g. McCarthy 2007). However, it is sufficient to define what we mean by AI in the present text: Here, the quest for AI is really the quest for strong- or general-AI, meaning AI that is comparable to (or exceeding) human intelligence on a complete spectrum of tasks or on so-called AI-complete tasks (Kurzweil 2005).
AI-complete tasks are tasks that require general-AI; an approach capable of handling one AI-complete task would be able to handle them all. AI-complete tasks include automatic speech recognition and machine translation if we require human-level performance.
Obviously, at this point, general-AI remains pure science-fiction, but unlike other interesting concepts from science-fiction, like teleportation, we actually have something concrete to go on: The brain is a reference system; the algorithmic principles that underlie the sought-after intelligence in human brains (to some extent in brains in general) should in principle be reusable in AI. The open question is how do we identify these principles and how can we apply them in AI?
This type of problem generally lends itself to the process of reverse-engineering:
The simplest form of reverse-engineering is basically taking a device physically apart, while carefully examining the parts, and from there inferring how it works to the point of reproducibility. However, here, and in many other cases, this is not really a feasible approach, because "taking the device apart" in this simple way is not reasonably possible.
We cannot simply take the brain apart to see how cognition works and we agree with the ideas of *embodied cognition* that cognition must be seen in very close connection with body and environment (see e.g. Wilson 2002). This means that we must go about the reverse-engineering process in a different way:
In a general-AI system we want to achieve functionality that a reference system, the brain, possesses. How should we go about this reverse-engineering task without taking the brain apart? We can observe different features of the brain; for most of these features we can ei-

ther come up with different ways to implement them or no way at all. In reverse-engineering these features are not particularly interesting, because they do not help us pick a development path and the odds of picking a path that reflects a key principle of cognition are not great. Conversely, the features that are interesting are the features that we only know of one (reasonable) way to implement, because if we want to implement them at all we must follow the only known path. This does not come with a guarantee of success; it might be that we only know of one way to implement something because we are ignorant of other possibilities, but we will probably not find better odds in terms of establishing a key principle. The crucial point is whether we are able to apply the principles of such a unique implementation of some specific feature to other features (that we might know of alternative ways to achieve) as well; if we are then this will significantly strengthen the principle's position as a key principle.

For example, one prominent feature of the brain is its very high data processing capabilities; we know that this is in large part achieved though highly parallel processing (Thagard 2005). Parallel processing as an algorithmic principle gives the brain computational speed. However, any computation that can be done in parallel can also be done on a serial system; if the serial system is fast enough it can fully simulate the parallel processes in relative real-time (McCarthy 2007). In the present reverse-engineering perspective this means that parallel processing is probably not a key principle in the sense that it does not directly restrict the possible development directions. While parallel processing can speed up computation, the fact that the same computations can be done another way (serially) suggests that implementing parallel processing is not likely to contribute anything else than speed to the AI problem. However, if we can find another characteristic brain feature that we only know of one single way to implement then this is a key principle, especially if it also brings additional insight to the AI problem. This will become clearer with an example:

Another very important feature of the brain is its robustness to physical damage; we know this is largely achieved by highly distributed neural representation (Thagard 2005). We might find different implementations of distributed representation from the AI side, but an implementation of the same robustness without distributed representation seems much more difficult.

This tentatively establishes distributed representation as a key principle; but to strengthen this idea we must look for additional contributions to the AI problem:

It happens that we do have a few different implementations of distributed representation in AI (Neumann 2001); one of the most interesting suggestions is based on the principles of

holography and is known as *holographic neural networks*. Is this in any way plausible, and does it provide any additional contributions to the AI problem?

Optical holograms made with lasers show a similar robustness; if a holographic recording medium is damaged the drop in intensity of the hologram is proportional to the area damaged, the hologram is not lost, but fades gracefully (this is because the information in the hologram is distributed) (Leith 1964). This is of course very different from traditional computer architectures where even slight (local) damage can easily crash the whole system.

The principles of holography do not only apply to the common laser-made holograms; it applies to waves in general through the laws of wave-interference. This includes waves propagating in a neural network medium; in fact, Westlake (1970) has shown that the basic holographic principles are possible in an excitatory postsynaptic potential (EPSP) neural network without much complication. Nonetheless, an implementation seems a daunting task, and fortunately a further generalization is possible through a mathematical analogy with circular convolution and de-convolution (other interesting holographic analogies are discussed in (Rabal 2001)).

The common term "holographic memory" is slightly misleading for this paradigm, because the focus on memory might neglect the cognitive aspects; in fact, the distributed representations can interfere with each other to produce strong generalization and association capabilities, like content-addressable memory; this kind of memory is not addressed one data-pointer at a time as conventional computer memory, but with a partial representation of the memory content we wish to retrieve. As we will see in the next section, content-addressability in holographic neural networks provides a natural approach to difficult AI issues like intuition. This is exactly the kind of additional contribution to the AI problem that we need in order to confirm distributed representation as a key principle.

An AI approach involving circular convolution does not sound immediately plausible, but through the analogy to holography it achieves higher plausibility both in terms of robustness and in the neural network perspective given by Westlake early on.

With the above considerations we have some concrete guidance on what should be considered important in our AI approach: Parallel processing is certainly interesting, but not as essential as distributed representation.

Let us see if we can establish further guidance from the reverse-engineering perspective:

Another outstanding feature of the brain is its remarkable flexibility; parallel processing and distributed representation of course both contribute, but we know that a lot of flexibility is also achieved through the brain's *neuroplasticity* (Mussa-lvaldi 2007). Is plasticity a key principle in the reverse-engineering perspective?

Let us limit ourselves to the kind of plasticity known as *synaptic plasticity*, which concerns the formation and destruction of synaptic connections between neurons as well as the strengths of these connections. A popular theory on explaining the workings of synaptic plasticity is *Hebbian learning*, which in popular terms is often phrased as "neurons that fire together, wire together" (in neuroscience this effect is also known as *long-term potentiation*). However, if we assume that neural representations are distributed then the Hebbian view does not necessarily answer the question of how the local plasticity "knows" what to do in a non-local, distributed perspective. We would expect the answer to be found within classical physics, but we do not know the answer and in a reverse-engineering spirit we might consider other suggestions. The following suggestion was conceived by the acclaimed mathematician and physicist Roger Penrose; we will not argue that it is necessarily plausible, but it will help raise the final brain issue to be considered here.

According to Penrose an analogous problem is how quasi-crystals grow; it seems that these structures can grow non-locally, which again seems difficult to explain in terms of classical physics. Instead the suggestion is to invoke quantum magic; the idea is that different, let's say, "growth-patterns" exist in quantum superposition until a certain energy level is reached and the system collapses into a single physical representation (see Penrose 1989). A similar non-local quantum process could be imagined for plasticity, which brings us to the real issue: Regardless of whether quantum effects are involved in plasticity the larger question is whether it is involved in brain processes at all. There does not seem to be any concrete experimental evidence in favor, but with our limited technical capabilities in this area, it might be allowable to rely as much on philosophical argument:

Let us return to the notion of AI-complete tasks for a moment; it is not entirely clear if very high-level processes, like consciousness and self, are required for general-AI. It is reasonable that for instance considerable social understanding is required in various AI-complete tasks, like machine translation, and it might be that social understanding cannot be achieved without consciousness and self.

These high-level features are also interesting in relation to the quantum question:

One of the defining features of consciousness and self is the feeling of free will; like the quasi-crystals this also appears to conflict with the determinism of classical physics, phi-

losophically we then have three options: 1) we can say something along the lines of that these high-level features *supervene* on the physical laws and *emerge* from dynamical feedback processes (Dennett 1991). 2) free will might be an illusion (Searle 1996). 3) some unknown quantum effects might be responsible (Penrose 1989).
Let us immediately disqualify option number two as its validity either way does not impact the current arguments. Option number one is quite popular and certainly not unreasonable, but also not a fact.
Penrose suggests an interpretation of Gödel's theorems that points more to door number three: Very briefly, a formal system can formulate statements that are true (that can be seen to be true), but cannot be proven within the formal system. This is taken to mean that since there can then be no algorithm to prove the truth, and humans on the other hand can see the truth, then our minds must be capable of non-algorithmic problem-solving.
If this is the case then we cannot hope to achieve general-AI, bar the advent of quantum computers or other unknown non-algorithmic computers.
On the other hand, if option number one is indeed correct then we should focus our research efforts on systems that are not only distributed, but also highly dynamical systems, i.e. systems can allow emergence of high-level features.
In any case, there is not much doubt that brain processes, even at the lower levels, are dynamic (i.e. with feedback) (Thagard 2005), but this alone does not necessarily establish it as a key principle; however considering that there are high-level features (free will) that we cannot achieve any other way (except with quantum magic) suggests that dynamical systems is indeed a key principle.

This adds dynamical systems to distributed representations (and partly parallel processing) as key research areas in our AI approach. Neuroplasticity and even quantum considerations were instrumental, but they are not considered key principles.

We can speculate that dynamical systems with distributed representation (and possibly, but not essentially, with parallel processing) qualify to take on AI-complete tasks, while a system with distributed representation alone might not be up to the challenge if high-level cognitive processing is required.

Finally, let us briefly consider where we stand in relation to other important approaches to AI: The classical *symbolic* AI approach primarily models cognition explicitly in terms of facts and rules. It has proven difficult (by the lack of practical examples) to produce sym-

bolic AI systems with enough explicit facts and rules to operate successfully outside a very narrow domain. To be fair this is arguably true for any AI approach so far. Symbolic AI is often contrasted with *connectionism*, which generally models cognition by networks of (interconnected) simple processing units, mostly known as *neural networks*. However, this is not the exact distinction we are making here; in fact, some connectionist approaches are *localist*, which means that although networks are used the representations in those networks are not highly distributed; instead representation is done by semantic nodes and relations between nodes such that each node has a (variable) activation-value and activation can spread between nodes through weighted relations. Memory search and inference in a localist network is therefore typically done by *spreading activation* (Anderson 1983).

The distinction we want to make here is between approaches with or without distributed representation; symbolic AI and the localist part of connectionism are called *structuralist*, while distributed connectionist approaches are called *componential,* following Hinton (1986). It should be noted that some approaches arguably use both localist and distributed representation, and in such cases the correct label is a judgment call.

Structuralist approaches include hidden Markov models, support vector machines, ACT-R (Anderson 1993, Anderson and Lebiere 1998) and the connectionist version ACT-RN (Lebiere and Anderson 1993), and many more.

A particularly interesting structuralist example is the semantic web vision as set up by the World Wide Web Consortium; this vision is based on the structuralist RDF triple data format (see Passin 2004), which is great for manually entering information (which might not even be desirable to do, because it is not statistically sensitive (see Doctorow 2001)), but is more limited when it comes to search and inference: Given a set of related RDF triples it is possible to reason over the data and thereby create new valid data. This can be called deductive reasoning build up of syllogisms (two triples used to deduce a third). A syllogism example could be:

> *The Semantic Web is made up of Syllogisms*
>
> *Syllogisms are not very useful*
>
> *Therefore, the Semantic Web is not very useful*

There are quite different expectations of the semantic web and of the level of AI it represents (Marshall and Shipman 2003), but in any case the syllogism example reflects an issue raised by Shirky (2003), namely that deductive reasoning alone is not powerful enough to enable general-AI. Also, with structuralist RDF there is no *efficient* way to implement content-addressable memory (unlike for componential approaches), which means that search problems can easily become intractable. Another difference from componential approaches

is that if RDF data is damaged relations between concepts are typically broken, which means that entire concepts can be un-retrievable.

In the next section we will see how issues like these become natural strong-points in a componential semantic web with distributed representation. We will focus on distributed representations in holographic neural networks; for reference other important distributed approaches are know as *tensor product representation* (Smolensky 1990), which uses an (expanding) outer product operation in place of circular convolution, *recursive auto-associative memory* (Pollack 1990), which uses a three layer network where the hidden layer learns distributed representations, and *binary spatter code* (Kanerva 1998), which uses high-dimensional binary vectors for which the XOR operation can be used in place of circular convolution. The motivation for choosing holographic neural networks over these alternative schemes is partly based on aesthetics (a factor that is not uncommon in science (Farmelo 2002)); more specifically it is that holography is based on a natural phenomenon (wave-interference), which the alternative approaches cannot claim.

# 3. Holographic Neural Networks

A key issue in any AI-approach is how to represent complex information (structured in hierarchies for example) in a way that can also be adequately processed by the AI system.
For most AI tasks a large part of the information in the domain can be modeled by hierarchies, tree-structures and similar networks of nodes.
In automatic speech recognition (ASR), for instance, speech information is often represented in networks of nodes at multiple levels of analysis; at a semantic level, at the syntactic level of sentence structure, and at lower levels of speech sounds, such as vowels and consonants. Concepts can be represented in class hierarchies for example and similar syntactic trees are often used for sentence structure.
This is a very intuitive approach, because we can read and understand these structures quite easily. However, if we are trying to reverse-engineer the brain (as specified in the previous section) we must also consider how these high-level structures can be mapped to plausible low-level neural representations.
A typical structuralist semantic network represents concepts as single nodes and relations as weighted connections, which can be learned from statistical information or even hand-made. In speech recognition this can for instance be used to help choose between recognition candidates, because the semantic network can suggest which words and concepts are related to the already recognized words, i.e. which words are more likely to occur in the context (e.g. Lieberman 2005).
On the other hand, we know that neither concepts, words nor vowels are represented in the brain as single nodes or neurons (Thagard 2005). More plausible representations would involve distributed patterns of activity across many neurons, but it is far from obvious how this kind of representation can connect to high-level structures; in fact it has long been widely held that distributed representations cannot usefully represent complex high-level data structures (Fodor and McLaughlin 1990), like hierarchies. However, at this point several distributed approaches have shown this expectation to be false (Gelder and Niklasson 1994a, Plate 1995, and other), but its counter-intuitive nature has meant that relatively few studies have been done with distributed representations. The feature that has been claimed to be missing in distributed approaches is often referred to as *systematicity* (Fodor and Pylyshyn 1988); we will return to this concept soon and see how a high level of systematicity can be achieved with distributed representations.

It is not enough to be able to represent complex data structures; the representations must also be processable. In a non-distributed structuralist semantic network processing is simply a matter of following connections from node to node, one node at a time. This reflects the conventional pointer-based computer architecture where one data-pointer is followed at a time. The processing requirements in this approach scales up with the number of nodes to be considered.
The situation is quite different for distributed representations; below we will, among other things, see that many nodes can be considered without directly processing each of them.

The following subsection will consider some specific aspects of the theoretical background of holographic neural networks: Geoffrey Hinton, Tony Plate and Jane Neumann have all contributed to a concrete and quite successful approach based on circular convolution and de-convolution as a mathematical analogy to holography. We will go through the major points for each contributor and consider a possible improvement.
Finally a simple HNN classifier will be introduced to test the improvement on automatic vowel recognition, and future directions for HNN for more difficult AI tasks will be discussed.

## 3.1 Theoretical Background

The theoretical background of HNN starts with Hinton's *reduced representations*; a derivative of distributed representations with a framework for representing and processing complex high-level structures with low-level distributed representations.

### 3.1.1 Hinton's Reduced Representations

Hinton (1990) analyzed the problem of representing complex hierarchical structure in distributed representations, and introduced the general concept of "reduced description" (or "reduced representation"). Reduced representations are powerful, because they can represent complex data structures in a distributed network, and allow fast operations on the data at the same time. To see how this works we must first consider the basic features of reduced representations.
The basic ideas of reduced representation go as follows:

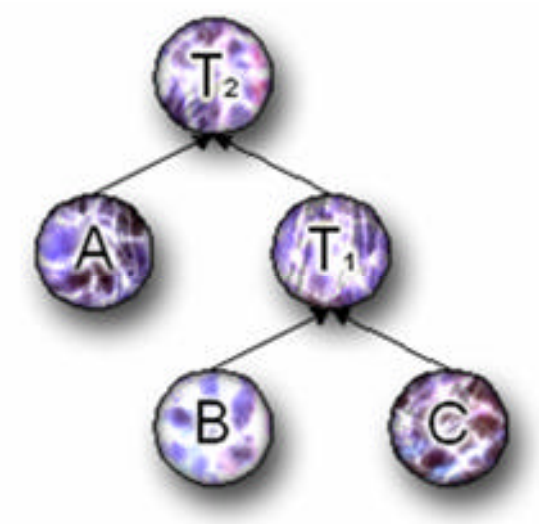


**Figure 1**: Partial concept hierarchy

With distributed representation concepts are patterns of activity over many network nodes, and each node can participate in multiple concepts. Consider the partial concept hierarchy in figure 1: This structure is simple to represent with traditional symbols, but with distributed representation it takes a bit more imagination; each concept (capital letters) is a distributed representation on a fixed-size vector (indicated by circles). $\mathbf{T_1}$ is a reduced representation of **B** and **C**, and $\mathbf{T_2}$ is a reduced representation of **A**, **B** and **C**. Moving up in the hierarchy is therefore compression, down is de-compression.

The two key points are:

1) The fixed size of reduced representations means that they can be used recursively without expanding the memory.

2) Reduced representations are different from traditional data pointers in the sense that a pointer itself is chosen arbitrarily and does not contain any information about the data it points to, while reduced representations maintain a meaningful reduced (compressed) version of the original data.

The second point opens up Hinton's notion of "rational inference" versus "intuitive inference": Intuitive inference is carried out on reduced representations without decompressing them into their constituents, while rational inference requires decompression. Intuitive inference is possible, because the reduced representations are not random, but reflect their constituents to some degree. This is of major importance because it affords a critical reduction in computational complexity of inference, for example, imagine that we have another structure identical to that in figure 1, but we do not actually know that the two structures are identical. To find out whether the two structures are the same we would normally (in an equivalent symbolic or localist representation) have to follow pointers to each element and compare element by element; the reduced representations on the other hand allow us to simply compare the most reduced descriptions ($\mathbf{T_1}$). This possibility for inference that is not just rational and step-by-step, but rather based on reduced representations of a larger context, is missing in regular semantic networks both when it comes to the type of inference and when we consider the computational advantages of not having to follow a much longer chain of rational inference.

Hinton's notion includes that concepts in "attentional focus" are decompressed allowing rational inference, while other concepts (not in attentional focus) remain compressed, but accessible through intuitive inference. If $\mathbf{T_1}$ is in attentional focus it is decompressed into **A** and $\mathbf{T_2}$, and even though **B** and **C** are not the focus of attention they are still accessible through intuitive inference on $\mathbf{T_2}$.

Although Hinton did do experiments, his concepts are more of a general framework with the essential compression and decompression operators open to different implementations; of the four schemes for distributed representation listed earlier (tensor product representation, recursive auto-associative memory, binary spatter code, holographic neural networks) it seems only one does not meet Hinton's requirements, namely tensor product representation (because it expands the memory resources, i.e. the vectors are not fixed-size).

### 3.1.2 Plate's Holographic Reduced Representations

Holographic reduced representations (HRR) introduced by Plate (1995) are an implementation of Hinton's reduced representations with circular convolution (denoted by ⊛) as the compression operator.

The circular convolution, $\mathbf{a} \circledast \mathbf{b}$, of discrete signals **a** and **b** is given by:

$$(\mathbf{a} \circledast \mathbf{b})_j = \sum_{k=0}^{n-1} \mathbf{a}_k * \mathbf{b}_{j-k}$$

For j = 0 to n-1 and where the circular effect is achieved by treating subscripts modulo n.

The holographic reduced representation, **T**, of **a** and **b** is given by:

$$\mathbf{T} = \mathbf{a} \circledast \mathbf{b}$$

Note that while the above expression has **a** and **b** on equal terms, in the HRR scheme one, say **a**, represents the data and the other (**b**) represents a reference key or ID, which is analogous to the reference beam used in holographic storage with lasers (Haw 2003).

Circular convolution can be computed by element-wise multiplication (denoted by .*) in the Fourier domain:

$$\mathbf{T} = \mathbf{ifft}(\ \mathbf{fft(a)} .* \mathbf{fft(b)}\ )$$

Another approach is to embed **a** in a right-circulant matrix, $\mathbf{[A]_r}$, of the form (also see figure 2):

$$(\mathbf{[A]_r})_{j,k} = (\mathbf{a}_{k-j})$$

where k, j = 0 to n-1 and subscripts are treated modulo n.

This allows the circular convolution to be computed by matrix-vector multiplication:

$$\mathbf{T} = \mathbf{[A]_r} * \mathbf{b}$$

The FFT version is faster and more so with more vector elements (Kvasnicka 2006), but as we will see below the matrix form has other advantages when it comes to developing a circular de-convolution procedure.

$$[A]_r = \begin{bmatrix} a_0 & a_{n-1} & \cdots & a_2 & a_1 \\ a_1 & a_0 & a_{n-1} & & a_2 \\ \vdots & a_1 & a_0 & \ddots & \vdots \\ a_{n-2} & & \ddots & \ddots & a_{n-1} \\ a_{n-1} & a_{n-2} & \cdots & a_1 & a_0 \end{bmatrix}$$

**Figure 2:** Right-circulant matrix

Holographic reduced representations can also be formed by superposition of other HRR. Circular convolution captures structured similarity, i.e. "A?B is similar to C?D to the extent that A is similar to C and B is similar to D" (Neumann 2001), while superposition captures unstructured similarity, i.e. the superposition of two vectors is similar to both. This means that structured information as in the hierarchy in figure 1 can be represented by convolutions and that multiple representations can be superimposed in the same memory.
The similarity of two HRR vectors is given by their dot product without decompressing and comparing individual constituents. In Hinton's framework this is an example of an intuitive inference; in more general terms (actually Neumann's terms as we will see shortly) it is a basic holistic transformation. This means that it is a trivial matter to compare different HRR for recognition, classification etc.

However, while HRR are useful even without decompression, for rational inference (and for learning more complex holistic transformations), decompression (i.e. circular de-convolution) remains an issue:
Circular de-convolution (denoted by ?) seeks to invert the circular convolution process as accurately as possible, but as in many other applications the exact inverse:

$$\mathbf{b} = \mathbf{a} \,?\, \mathbf{T} = [\mathbf{A}]_r^{-1} * \mathbf{T}$$

is not a particular good option, because the inverse is sensitive to noise, i.e. noise in the data becomes amplified by the inverse operation (Plate 1991). Also, the exact inverse might not exist; this also goes for the FFT based version of circular de-convolution, which is simply element-wise division in the Fourier domain. If the exact inverse does not exist we would like an approximation:
In Plate's scheme, circular de-convolution is approximated by circular correlation. The circular correlation, $\mathbf{a} \,?\, \mathbf{T}$, of **a** and **T** is given by:

$$(\mathbf{a} \circledast \mathbf{T})_j = \sum_{k=0}^{n-1} \mathbf{a}_k * \mathbf{T}_{k+j}$$

For j, k = 0 to n-1 and subscripts modulo n.

As a matrix-vector multiplication expression we must first create the involution **ia** of **a**, simply:

$$\mathbf{ia}_i = \mathbf{a}_{-i} \quad \text{,again modulo n}$$

With circular correlation instead of the exact inverse the de-convolution expression then becomes:

$$\mathbf{b'} = \mathbf{a} \circledast \mathbf{T} = [\mathbf{IA}]_r * \mathbf{T}$$

However, circular correlation only approximates the exact inverse for a certain class of vectors, called noise-like vectors that must also be high-dimensional. Typically a normal distribution, N(0,1/n), is used. The larger the vectors (typically 4096 elements), the better the de-convolution approximation, i.e. the interpretation of the operation is that it seems simpler data structures are harder to handle.

This restriction to noise-like vectors means that for practical use (where features are typically not noise-like) signal feature-vectors must be mapped to noise-like feature-vectors. This comes with a computational cost of course and the mapping itself must also be managed. An additional complication is that the decompressed features **b'** (retrieved from the memory **T** by circular correlation) only makes sense to the extent that they can be mapped back to real signal features. It is unlikely that the de-convolution is perfect (in the sense that **b' = b**), so a clean-up memory must be used for accurate reconstruction of **b** (and thereby of actual signal features). The clean-up memory procedure simply picks the highest dot product of the decompressed vector **b'** and elements in the clean-up memory. However, if we want to map **b'** back to signal features that are not explicitly in the clean-up memory we have to use a more advanced clean-up memory that is able to generalize over mapping examples.

The simple HNN classifier to be introduced shortly will try to address these issues following a suggestion by Schönemann (1987) with a different de-convolution procedure. First, let us take a look at how HRR have been used in recurrent neural networks.

### 3.1.3 Recurrent Holographic Neural Networks

Recurrent networks are networks with feedback connections, which mean they are dynamical systems that can exhibit emergent properties. As discussed in the previous section emer-

gence from dynamical processes is a strong philosophical position in explaining very high-level cognitive features and there is also not much doubt that the brain is highly dynamical. We will not be implementing a recurrent network for AVR, but because we want our efforts to eventually go in the more difficult ASR-task direction, we want to consider HRR in relation to recurrent networks.

In the present context recurrent holographic neural networks are recurrent networks based on holographic reduced representations (HRR).

Although HRR have many useful properties by themselves, more advanced tasks, like ASR, or simply tasks involving sequences probably require a solution more sensitive to temporal dynamics; figure 3 shows a basic recurrent HNN: An input sequence $I_i$ = a, b, c is convolved with its context, $T_{i-1}$ (a HRR of previous sequence elements), to form the next HRR, this HRR then becomes the next context. The representation or encoding of the input sequence becomes:

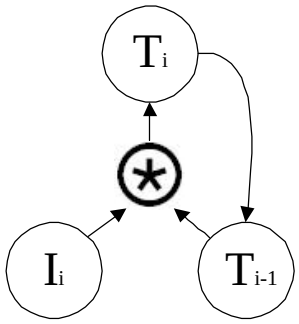



**Figure 3:** Simple recurrent HNN, input sequence $I_i$, HRR $T_i$, HRR context $T_{i-1}$.

$$\mathbf{T_n = a + a ⊛ b + a ⊛ b ⊛ c}$$ **,encoding of sequence**

This HRR can be decoded by de-convolving with the HRR of the sequence leading up to the element to be retrieved, e.g. $\mathbf{T_n}$ de-convolved with $\mathbf{a ⊛ b}$ gives **c** (after clean-up memory).

Another variation uses a "method of loci"-style "trajectory-association" where each sequence element is convolved with different points along a predetermined trajectory.

Plate (1993) employs a trajectory determined by successive "convolutional powers" of a noise-like vector, **k** (see Plate (1993) for the definition):

$$\mathbf{T_n = a + b ⊛ k^1 + c ⊛ k^2}$$

This approach was used to successfully compare recurrent networks with HRR to simple recurrent networks (SRN) on sequence generation tasks, in one case pen trajectories of handwritten digits.

Another example of recurrent HNN is found in (Astakhov 2007), which claims to combine HRR with Adaptive Resonance Theory (ART) (Grossberg 2003) to produce an imagination simulation system called "Script Writer". One highlight being that concepts in memory are re-categorized each time a concept is re-experienced, e.g. the concept "cat" is revised each time a cat is seen.

In general, recurrent networks are often trained by back-propagation through time (unfolding in time method) or real-time recurrent learning, but other methods have also been tried. See for instance (Schiller 2005) for a comparison of recurrent training methods.

Several comments have been made on HNN in the literature; the bulk of the considerations are in the context of the long-standing dispute between symbolist and connectionist viewpoints (e.g. Eliasmith 1997, Thagard 2001) and do concern themselves with the recurrence aspects of HNN. Levy (2006) takes a more forward-looking perspective and suggests the combination of HRR and self-organizing maps (SOM). A highlight is that the SOM approach allows unsupervised learning in high-dimensional space. Levy's tentative conclusions are that HRR can indeed both represent distributed low-level neural data and high-level data with complex structure, but to create systems that can exploit this on hard AI-tasks we will need new organizational principles, like SOM and fractals (Levy 2007).

Another related approach that could possibly be combined with HRR is known as echo state networks (ESN) (Jaeger 2001). This is also a recurrent type network; strongly related to liquid state machines (LSM) (Maass 2002) – the main difference being that LSM usually focus on more plausible neuron models. On the other hand, even a quantum LSM has been suggested (Herman 2007) and ESN with quite realistic neurons have also been created.
ESN and LSM are reservoir computing techniques, hence the term "liquid", and they rely on dynamics of the reservoir that behave generally according to wave-interference. This could potentially be combined with holographic principles, since the main prerequisite is indeed wave-interference.
Next, we will consider how HNN that are not recurrent can learn to generalize with so-called holistic transformations.

### 3.1.4 Neumann's Holistic Transformations and One-Shot Learning

As already seen, holistic transformations are transformations that operate on HRR without decomposing them. We can get an understanding of what holistic transformations can do by considering their possible level of systematicity (ability to generalize); as mentioned earlier, the claim that distributed systems do not allow a high level of systematicity has been refuted. Systematicity is basically a measure of a system's ability to generalize from training data to unseen test data. The systematicity scale proposed by Niklasson and van Gelder (1994) has five levels (with five as highest); the definition below is rephrased in Neumann's (2001) words:

*Level 0* *No generalisation. The system only remembers the training examples.*

*Level 1* *Generalisation to novel composed structures. The constituents of the test structures appear in all their syntactically allowed positions during training. The system only generalises to new combinations of constituents.*

*Level 2* *Generalisation to novel positions of constituents. The training set contains*

*all constituents of the test structures but not all of them in all their syntactically allowed positions.*

*Level 3* *Generalisation to novel constituents. Some constituents of the test structures do not appear in the training set.*

*Level 4* *Generalisation to novel complexity. Some test structures are of higher complexity than the structures used for training.*

***Level 5*** ***Generalisation to novel constituents in structures of higher complexity. Some test structures are of higher complexity than the structures used for training and contain constituents that did not appear in the training set.***

Holistic transformations on HRR with level five systematicity have been demonstrated by Neumann (2001) on propositional logic tasks: *"Our system generalised acquired knowledge about the transformation of hierarchical structures to structures containing novel elements and to structures of higher complexity than seen in the training set. This corresponds to Level 5 systematicity as defined by Niklasson (1993), which, to our best knowledge, has not been achieved by any other comparable method."*

Level five systematicity is a quite remarkable success that comes close to human-level capabilities; for example when we hear a new word for the first time in a sentence we are usually able to immediately use the new word in other sentences even with the word in other forms. This is an example of level five systematicity.

Nevertheless, in many AI tasks systematicity alone is not enough, multiple transformations would be needed and the system would have to actively manage the different transformations. In Neumanns words: *"We believe that transformations of this kind could provide a means for efficiently solving more complex problems that require a high degree of systematicity, such as logical inference. However, performing a chain of inference clearly involves more than a single structural transformation of logical expressions. A number of different transformation rules have to be acquired and appropriately applied by the system." (Neumann 2001)*

Recurrent networks probably hold part of the solution, but additional issues like attentional focus will likely have to be worked out as well, and there are so far no concrete suggestions for a more complete AI system based on HNN.

A part of Neumann's success lies in the introduction of a clever approach to learning holistic transformations from examples. The approach is much simpler and performs better than for instance back-propagation. It was named "one-shot learning", because it is achieved by a single pass through all training data: We want to learn a transformation vector, **T**, which when convolved with input, **A**, gives correct output, **B**. From the observation that the gradient of the error function should be zero, Neumann obtained:

$$\underbrace{\sum_{i=1}^{m}(\mathbf{B}_i \oplus \mathbf{A}_i)}_{\mathbf{V}} = \underbrace{\sum_{i=1}^{m}(\mathbf{B}_i \oplus \mathbf{B}_i) \circledast \mathbf{T}}_{\mathbf{U}}$$

Where **i** runs through all input-output pairs.

This simply gives the desired transformation vector as:

$$\mathbf{T} = [\mathbf{U}]_r^{-1} * \mathbf{V}$$

The learned transformation vector, **T**, has the ability to generalize to level five systematicity as stated above. This probably exceeds what is needed for AVR, but we do need the ability to learn to generalize over vowel examples, so this learning scheme is a natural approach for AVR with HNN.

The only adjustable parameter is the dimension of the noise-like vectors, which is not actually a real parameter, since its meaning it simply that the higher the dimension the better the results. For vectors that are not high-dimensional the de-convolution by correlation approximation does not hold and the scheme breaks down. If we can overcome this restriction to high-dimensional noise-like vectors we will speed up an already very simple learning scheme (by the amount that we can reduce dimensionality) and there would also not be a need for an intermediate mapping between real signal features and noise-like features.

Next, we will develop a new HNN classifier based on Neumann's one-shot learning scheme, but without the restriction to noise-like high-dimensional features.

## 3.2 A Simple HNN Classifier

The simple HNN classifier that we will develop next aims at overcoming the restriction to noise-like feature vectors imposed on the HNN systems described above.

Schönemann (1987) suggested that the Moore-Penrose pseudo-inverse has several advantages over both the exact inverse and the correlation approximation used for circular de-convolution by Plate. A highlight is indeed that feature vectors need not be noise-like.

Below we will develop an effective implementation of the pseudo-inverse that can be accelerated by parallel processing on a GPU. Then we will adjust Neumann's one-shot learning approach to the new pseudo-inverse de-convolution procedure.

### 3.2.1 Moore-Penrose Pseudo-Inverse De-convolution

Let us first introduce the relevant expressions representing Schönemann's idea to replace the circular correlation with the pseudo-inverse (indicated by $^+$) in the de-convolution step.

The circular convolution is still given by:

$$\mathbf{T} = \mathbf{a} ? \mathbf{b} = [\mathbf{A}]_r * \mathbf{b} \qquad \text{,circular convolution}$$

The pseudo-inverse $\mathbf{[X]_r^+}$ of $\mathbf{[X]_r}$ is a unique matrix that per definition satisfies the following criteria:

1. $\mathbf{[X]_r[X]_r^+[X]_r = [X]_r}$
2. $\mathbf{[X]_r^+[X]_r[X]_r^+ = [X]_r^+}$
3. $\mathbf{([X]_r[X]_r^+)^* = [X]_r[X]_r^+}$
4. $\mathbf{([X]_r^+[X]_r)^* = [X]_r^+[X]_r}$

Where $^*$ is the conjugate transpose (or the transpose for real-values matrices).
For any problem of the form:

$$\mathbf{T = [X]_r * b}$$

the shortest length least squares solution is:

$$\mathbf{b = [X]_r^+ * T}$$

The pseudo-inverse de-convolution from above then simply becomes:

$$\mathbf{b' = a \text{ ? } T = [A']_r^+ * T} \qquad \mathbf{\text{,circular de-convolution with pseudo-inverse}}$$

If **A' = A** then **b' = b**, for other cases **b'** is a least squares approximation.
In the following we will assume **A' = A** for simplicity.

Schönemann's suggestion leaves the question of how to effectively compute the pseudo-inverse and the defining criteria above do not provide an algorithm. We need a pseudo-inverse implementation that is faster than the standard Matlab implementation. Below we will detail our pseudo-inverse computation approach for speed-up with GPU.

### 3.2.2 Moore-Penrose Pseudo-Inverse Computation

Let us start with the convolution expression (switching left and right sides of the expression compared to earlier):

$$\mathbf{[A]_r * b = T} \qquad \mathbf{\text{,circular convolution}}$$

We are always allowed to multiply on both sides with the same factor; here we multiply with the transpose of $\mathbf{[A]_r}$ for reasons that will become clear momentarily:

$$\mathbf{[A]_r^T * [A]_r * b = [A]_r^T * T} \qquad \mathbf{\text{,multiply by transpose, } [A]_r^T \text{, on both sides}}$$

Then rearrange this to isolate **b** so that we can compare with $\mathbf{b = [A]_r^+ * T}$ (from above):

$$\mathbf{b = ([A]_r^T * [A]_r)^{-1} * [A]_r^T * T} \qquad \mathbf{\text{,compare this with } b = [A]_r^+ * T}$$

From the comparison we get an expression for the pseudo-inverse:

$$\mathbf{[A]_r^+ = ([A]_r^T * [A]_r)^{-1} * [A]_r^T} \qquad \mathbf{\text{,iff } ([A]_r^T * [A]_r)^{-1} \text{ exists}}$$

The effect of multiplying with the transpose is that the eigenvalues of $\mathbf{[A]_r^T * [A]_r}$ are either positive (square of $\mathbf{[A]_r}$'s eigenvalues) or zero (if the matrix is not invertible). To deal with the non-invertible case and obtain a more practical computation of $\mathbf{([A]_r^T * [A]_r)^{-1}}$ we tum to the pseudo-inverse again:

$$\mathbf{[A]_r^+ = ([A]_r^T * [A]_r)^+ * [A]_r^T}$$

Since $\mathbf{[A]_r}$ is a circulant matrix, so is $\mathbf{[A]_r^T * [A]_r}$ **,** and it is therefore (from the convolution theorem) diagonalized by the discreet Fourier matrix, **[F]**:

$$\mathbf{[E] = [F] * [A]_r^T * [A]_r * [F]^T}$$

Where **[E]** is a diagonal matrix of eigenvalues of $\mathbf{[A]_r^T * [A]_r}$

Diagonalization can be undone by reversing the expression like so:

$$\mathbf{[A]_r^T * [A]_r = [F] * [E] * [F]^T}$$

We then obtain the pseudo-inverse of $\mathbf{[A]_r^T * [A]_r}$ by (pseudo) inverting **[E]:**

$$\mathbf{([A]_r^T * [A]_r)^+ = [F] * [E]^+ * [F]^T}$$

Note that **[E]** is pseudo inverted by inverting each element of the diagonal, except very small (thresholded) eigenvalues, which are replaced by zeroes in $\mathbf{[E]^+}$.

We believe that discarding the smallest eigenvalues in this way is equivalent to the regularization technique of truncated singular value decomposition (TSVD) (see e.g. (Hansen 1987)). Cheng (1997) suggested using the technique to improve the numerical stability of transform based de-convolution after studies by Linzer (1992). Hansen (1996) has shown that the smallest eigenvalues mainly represent noise, which he has exploited in so-called *rank reduced noise reduction*.

Cheng (1997) also suggested reducing the computational complexity by using the real-valued Hartley matrix, **[H],** instead of Fourier. The convolution theorem also holds for some Fourier-related transforms, like the Hartley transform, and the same diagonalization applies:

$$\mathbf{[H] = [H]^{-1} = [H]^T}$$ **,simplifying identity**

$$\mathbf{[E] = [H] * [A]_r^T * [A]_r * [H]}$$ **,diagonalize**

$$\mathbf{([A]_r^T * [A]_r)^+ = [H] * [E]^+ * [H]}$$ **,pseudo invert and un-diagonalize**

$$[A]_r^+ = [H] * ([H] * [A]_r^T * [A]_r * [H])^+ * [H] * [A]_r^T \quad \text{,complete pseudo-inverse expression}$$

$$b = [A]_r^+ * T \quad \text{,de-convolution with pseudo-inverse}$$

This yields an effective circular de-convolution procedure, lending itself to straight-forward implementation through matrix-vector and matrix-matrix multiplication, which parallelize well onto the GPU (see appendix A). On our system we achieved a speed-up of an order of magnitude on one matrix-matrix multiplication of 2048x2048 elements from CPU to GPU; both multiplications done from within Matlab (compare with appendix A). On the entire pseudo-inverse step we achieved an order of magnitude speed-up over Matlab's pinv pseudo-inverse function at 1024 (x1024) elements; this speed-up was not only due to the GPU, but also because pinv does not take advantage of the matrices being circular.

The approach might also be relevant for other inverse problems; however this will not be explored here. It seems likely that further study of the above procedure could reveal an FFT based equivalent and thereby reduce the number of matrix-multiplications; this would be expected to provide additional speed-up.

Next, we will adapt Neumann's one-shot learning scheme to the new de-convolution procedure.

### 3.2.3 Adapted One-Shot Learning and Holistic Transform

Learning from examples has been described as an ill-posed inverse problem approachable with regularization techniques (Rosasco 2004), so the procedure developed above might be applicable to this scheme; however the scheme is not yet fully developed (De Vito 2005). Instead we choose to stay within the HNN background and adapt Neumann's one-shot learning scheme:

*Supervised learning* in the simple HNN classifier proceeds as follows: Each class is represented by a random noise-like vector, $\mathbf{B_{class}}$, as in the original scheme. For each class we learn a transformation vector, $\mathbf{T_{class}}$, by adding up de-convolutions of input, $\mathbf{A_n}$, and respective class vectors, $\mathbf{B_{class}}$ (here ? denotes circular de-convolution, not specifically circular correlation):

$$T_{class} = \sum_n A_n \oplus B_{class}$$

Where **n** runs through the elements belonging to the respective class.

The de-convolution is performed with the Moore-Penrose pseudo-inverse as described above. This scheme has one parameter; the truncation threshold in the inversion of the diagonal eigenvalues, which for small values should represent noise as already mentioned.
The same supervised learning of the transformation vectors, $\mathbf{T}_{class}$, is shown in pseudo-code below:

**for each class, $\mathbf{B}_{class}$**
    **for each feature vector, $\mathbf{A}_n$**
        **$\mathbf{T}_{class}$ += $\mathbf{A}_n$ ? $\mathbf{B}_{class}$**

*Classification* proceeds as follows: The unseen input, $\mathbf{A}_n$, of unknown class is convolved with each of the class transform vectors, $\mathbf{T}_{class}$, to produce $\mathbf{B'}_{class}$ (one **B'** for each class). The **B'** that is most similar (highest dot-product) to its respective $\mathbf{B}_{class}$ is chosen as the correct class for $\mathbf{A}_n$.
The pseudo-code to classify an unseen trial, $\mathbf{A}_n$, is listed below:

**for each transformation vector, $\mathbf{T}_{class}$**
    **$\mathbf{B'}_{class}$ = $\mathbf{A}_n$ ? $\mathbf{T}_{class}$**
**class := max(dot($\mathbf{B'}_{class}$, $\mathbf{B}_{class}$))**

This completes the simple HNN classifier.
De-convolution is computationally more expensive than convolution, so it is advantageous to have the de-convolutions in the learning phase and not the other way around.
It is to some extent possible to add up the transform vectors to create a single transformation vector (to further speed up classification); in this case the different classes will interact to the extent they are similar, and the dot products will be lower.

### 3.2.4 Discussion

The simple HNN classifier is not recurrent and it does not use a chain of holistic transformations as suggested by Neumann (in this sense it is a single holistic transformation), so per the earlier discussion it is probably not be up to an AI-complete challenge. This is reflected in the fact that we are attempting AVR and not ASR.
A very attractive feature of the new HNN classifier, compared to Plate and Neumann's approaches, is that feature vectors need not be noise-like. This means that, for example, spectral feature vectors can be used directly.

Further, feature vectors need not be very high dimensional (as the typical 4096-element vectors); although there must be enough dimensions to adequately distinguish between classes, the new de-convolution procedure itself does not depend on high dimensional vectors.

In the next section we will take a closer look at the AVR task, possible features and available datasets.

# 4. Automatic Vowel Recognition as an entry-level AI Task

The automatic vowel recognition (AVR) task is a sub-task of automatic speech recognition (ASR). While ASR is likely an AI-complete task (Shapiro 1992) requiring semantic understanding of the speech being recognized for human-level performance, AVR is an entry-level task, which does need the ability to generalize, but is more independent of the context. This does not imply that human vowel recognition is also context independent, but rather that the AVR task is independent of context by design in order to limit the task. Specifically, no real context is provided: A number of different single words are recorded by different speakers, typically in a consonant-vowel-consonant setup, like “had”, “hod”, “heed”. The AVR task is simply that, given a number of speakers for training, the system must classify vowels of unseen speakers. Traditionally, ASR and AVR systems are seen as two distinct parts; a feature extraction step and a classification step.

We have chosen to attempt AVR instead of ASR, because the simple HNN classifier developed above is probably not up to the ASR challenge for lack of recurrence (as discussed in the previous section). However, AVR and ASR can be done with similar kinds of spectral features, so if the simple HNN classifier can successfully interface with these features (without the previously necessary noise-like mapping) on the AVR task then it is also step in the right direction for the ASR task.

Secondly, AVR requires a large part of the same generalization abilities in the classifier as ASR does. A good result on AVR is therefore a reasonable prerequisite before attempting ASR.

In the following two subsections we will take a closer look at candidate feature extraction techniques and briefly consider traditional vowel classification. Then we will turn to actual datasets for AVR.

## 4.1 Feature Extraction

The general goal of feature extraction is to reduce the dimensionality of the data while maintaining the relevant information. Statistical measures and techniques like principal component analysis (PCA) or independent component analysis (ICA) can be used to deduce a few very discriminative features, but with the brain as a reference system it is natural to consider feature vectors that are modeled more according to our (peripheral) perception. Here PCA and ICA are not really plausible enough.

Traditional vowel studies have focused on very few features usually given by the fundamental frequency and spectral peaks called formants (usually four features in total), while ASR typically use cepstral features (a cepstrum is basically a spectrum of a spectrum), which are somewhat more involved.
In the following three subsections we will focus on some general techniques that are arguably appropriate for perceptual features for both AVR and ASR tasks.

### 4.1.1 Mel-Frequency Cepstral Coefficients

Mel-frequency cepstral coefficients (MFCC) are probably the most used features for automatic speech recognition. MFCC features are based on the Mel-scale which models the auditory system to some extent. It is very similar to the Bark-scale which is based on an estimation of critical bandwidths in the auditory system. Both scales are nearly linear at lower frequencies and approximately logarithmic at higher frequencies.
Creating an MFCC representation proceeds as follow:

1. Take the absolute value of the short-time Fourier transform (STFT).
2. Warp to Mel-scale with triangular overlapping windows.
3. De-correlate (remove redundancy and reduce dimensionality) by taking the discrete cosine transform (DCT) of the log-Mel-spectrum, and return first N components

Typically the first 13 components and 1st and 2nd derivatives are concatenated to form a 39-dimensional feature-vector for ASR.
A number of Matlab implementations are available.

### 4.1.2 Constant-Q Transform

The constant-Q transform suggests an improved time-frequency resolution over the STFT; Blankertz (199?) gives a nice introduction: *"The constant Q transform as introduced in [Brown, 1991] is very closely related to the Fourier transform. Like the Fourier transform a constant Q transform is a bank of filters, but in contrast to the former it has geometrically spaced center frequencies […]. This yields a constant ratio of frequency to resolution […]. What makes the constant Q transform so useful is that by an appropriate choice for f0 (minimal center frequency) and b the center frequencies directly correspond to musical notes.[…]. Another nice feature of the constant Q transform is its increasing time resolution towards higher frequencies. This resembles the situation in our auditory system. It is not only the digital computer that needs more time to perceive the frequency of a low tone but also our auditory sense".(p.1)*
To get a better feeling for this time-frequency resolution tradeoff consider figure 4, which gives a rough comparison of the time-frequency plane tilings of short-time Fourier transform (STFT), constant-Q transform and representative wavelet transforms.

The naïve computation of the constant-Q transform is expensive; however Brown and Puckette (1992) have shown that most of the expense can be written-off by applying a pre-computation step. The pre-computation step creates a transformation matrix corresponding to a bank of Fourier filters with variable windows.

Blankertz (199?) provides a Matlab implementation of this technique. Another Matlab implementation is available on Judith Brown's website (Brown 2008)

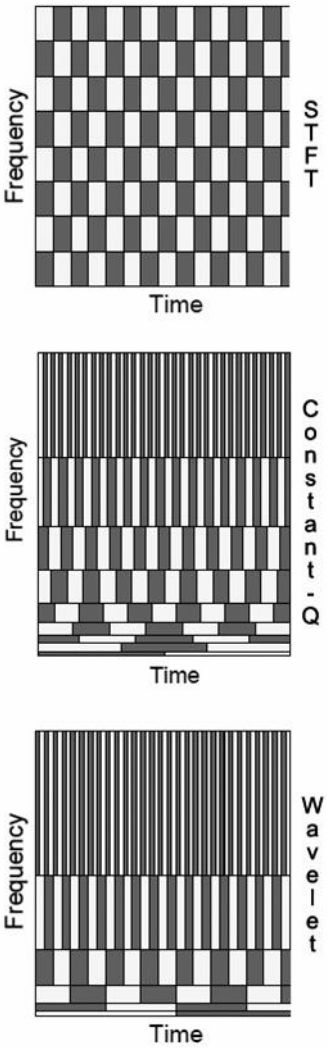



**Figure 4:** Rough comparison of TF-plane tilings. Adapted from Chatterji 2003.

Blankertz (1999) proceeds to suggest an invariant extension to the constant-Q transform. He defines the problems:

*"1. Owing to spectral leakage the magnitude of a constant Q transform (as well as the Fourier transform) is sensitive to phase changes in the stimulus, a phenomenon also termed 'Fourier uncertainty'. For employing pattern recognition methods a spectral representation is desirable that is invariant under phase changes.*

*2. We have the usual dilemma of trading time against frequency resolution."*

The invariant solution addresses these two problems (note that a "peak list" is a list of spectral intensity peaks): *"The proposed robust constant Q spectrum is the abstraction of such a peak list to a vector whose components correspond to fixed frequencies that are equally spaced in the log frequency domain. This vector looks formally like the usual constant Q transform but it has two advantages over the latter. (1) All nonzero values in that vector stem from actual peaks in spectral intensity and (2) the frequency information is not phase sensitive and it is reliable even at a high time resolution, i.e. the two problems discussed above are resolved."* (Blankertz 1999)

An invariant constant-Q transform is therefore very attractive for recognition and classification tasks, however the actual implementation was not found this time around.

### 4.1.3 Wavelet transforms

Feature extraction based on wavelet transforms is the most modern approach of the three considered here. A number of studies (e.g. Deviren (2003), Siafarikas (2005), Mporas (2007)) have shown that wavelet transform based techniques are generally superior to MFCC on ASR related tasks. It has also been shown that wavelets can provide better decorrelation than the DCT in the final step of the MFCC procedure (Sarikaya 1998).

One common technique is called *wavelet packet analysis*, in which the frequency content of the signal is represented by a binary tree-structure with wavelet packets at each node. It is possible to select packets from the tree at different frequency resolutions and create a Mel-scaled perceptual representation. Figure 5 shows different frequency resolutions at each level of the tree. The packets in solid bold lines represent a perceptual spectrum based on critical bandwidths, like the Bark-scale, but with an updated estimation of the required frequency resolutions at each band (see Siafarikas 2004).

The wavelet packet approach is so flexible that it can provide Mel-scaled representation as well as constant-Q representation (Long 1996).

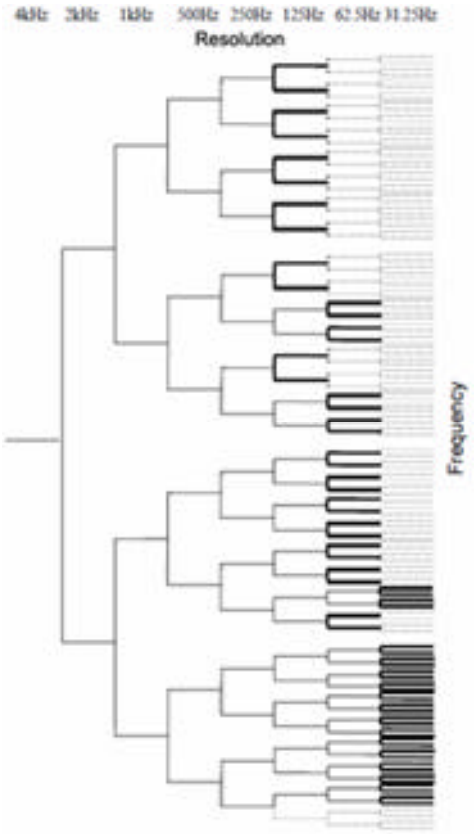



**Figure 5:** Wavelet packet tree. The packets in bold lines form a critical band based perceptual representation. From Siafarikas (2004).

Several invariant versions of the wavelet approach exist; all at the cost of additional computational complexity. These can be found as "stationary wavelet transforms" in Matlab documentation for instance, and other implementations are known as dual-tree or invariant wavelet transforms (e.g in Stanford's WaveLab package).

Another interesting aspect of wavelet transforms is their usefulness in compression, e.g. fractal image compression, which used to give blocky results (visually), has been generalized in a wavelet framework (see Davis 1998) to enable much smoother results by simply switching to another wavelet type. Perhaps this could be a useful approach to Levy's (2007) suggestion for fractal organizational principles mentioned in the previous section.

## 4.2 Classification

Most AVR systems use supervised training. A range of structuralist AI and machine learning techniques have been tried, including hidden Markov models, support vector machines and various artificial neural networks.

Recurrent systems have not yet made much of an impact in the AVR area however. This might be because recurrence is not critically needed for the AVR task (probably unlike the ASR case), and generally classifiers do pretty well even without recurrence (as we will see in the next section).

AVR systems are usually tested on unseen data, however in a few cases results have been reported on n-fold cross-validation (see e.g. Mitchell 1997), i.e. not entirely unseen samples if each speaker provides more than one sample.

A common issue for most classifiers is the dimensionality of the feature vectors; incoming data must be reduced to smaller feature vectors with the same information. Above we saw three examples of this process, and there are many more. Typically feature vector dimensionality is rather low, like the 39-element vectors from MFCC. This keeps the computational load down in the classifier, but mostly at the cost of additional complexity in the feature extraction and some loss of information. For some classifiers higher dimensional feature vectors are also difficult for other reasons than the computational load, like an increased risk of overfitting (Mitchell 1997), and it is generally hard to establish feature and dimensionality guidelines that are not simply based on what is convenient for a given classifier.

## 4.3 Datasets for Automatic Vowel Recognition

There is some degree of scarcity when it comes to actual vowel recognition reference datasets; often the raw audio is not available or the reference results are for vowel-based speaker identification and not directly vowel recognition.

We will briefly consider five vowel datasets. The first three datasets do not include the original audio, and are therefore not directly suited for testing the simple HNN classifier (because representation is given as very low dimensional non-perceptual features). However, they should give an impression of the diversity among vowel recognition results and indicate some common trends and issues. The final two datasets considered below (and also used later in testing the simple HNN classifier) will be placed in the context of the former three reference datasets.

| *Classifier* | *no. of hidden units* | *no. correct* | *percent correct* |
|---|---|---|---|
| Single-layer perceptron | - | 154 | 33 |
| Multi-layer perceptron | 88 | 234 | 51 |
| Multi-layer perceptron | 22 | 206 | 45 |
| Multi-layer perceptron | 11 | 203 | 44 |
| Modified Kanerva Model | 528 | 231 | 50 |
| Modified Kanerva Model | 88 | 197 | 43 |
| Radial Basis Function | 528 | 247 | 53 |
| Radial Basis Function | 88 | 220 | 48 |
| Gaussian node network | 528 | 252 | 55 |
| Gaussian node network | 88 | 247 | 53 |
| Gaussian node network | 22 | 250 | 54 |
| Gaussian node network | 11 | 211 | 47 |
| Square node network | 88 | 253 | 55 |
| Square node network | 22 | 236 | 51 |
| Square node network | 11 | 217 | 50 |
| Nearest neighbour | - | 260 | 56 |

**Table 1:** Typical machine classification results. From Robinson 1989.

### 4.3.1 Deterding's "Vowel" Dataset

The Vowel dataset (Deterding 1998) is a classical dataset of 11 English vowels. All vowels are represented by ten LAR features (log-area-ratios, the precise definition is not important here) computed from the original audio, effectively fixing the feature extraction step to one type of feature vector, which makes it straight-

forward to compare different classifier without worrying much about differences in features. Human listening results are unfortunately not available, so classification results cannot be directly compared to human performance.

Robinson (1989) made an extensive comparison with different classifiers (see table 1), and a similar study has been done with the WEKA machine learning framework (Klautau 2008). These results top out at around **55**% correct. Later studies have reported results around **70**% correct with support vector machines among other approaches (Ganapathiraju 1998).

The Vowel dataset is a difficult dataset, which we will see when comparing to the following datasets.

### 4.3.2 Peterson and Barney's "PBvowel" Dataset

The PBvowel dataset is also a classical vowel dataset. It contains ten English vowels represented by four features. The four features are estimates of the fundamental frequency and three spectral peaks (formants).WEKA results top out at around **87**% correct compared to **94**% for humans. The experimenters comment on the performance increase over the Vowel dataset: "*Besides being based on different English accents, the vowel and pbvowel databases were not obtained through the same experimental procedures. Also, formants can be seen as a more efficient representation of vowels than LAR parameters.*" (Klautau 2002)

We will return to the latter point about the effectiveness of the feature representations shortly.

Another interesting aspect of this dataset is that human performance is not very stable across the different vowels; especially two vowels are confused more often. The original human listening study comments: "*The very low scores of AA and AO result primarily from the fact that some members of the speaking group and many members of the listening group speak one of the forms of American dialects in which AA and AO are not differentiated*." (Klautau 2002)

This says something potentially important about the nature of the AVR task. Despite poor performance on some vowels the subjects are presumably still able to distinguish the different words in daily conversation, because of the additional context. This pin-points an important difference from the ASR task, i.e. the lack of any considerable context, which actually makes the AVR task harder than might be expected.

The characteristic AA/AO confusion trend might reasonably also be found in a plausible machine approach. Figure 6 shows confusion matrices for humans and the best WEKA classifier, the authors note: *"The pattern of errors is not exactly the same, but there are coincidences as, for example, vowels IY and ER are recognized with high accuracy in both cases. In relative terms, the machine has more troubles with UH, while listeners have with AA."* (Klautau 2002)

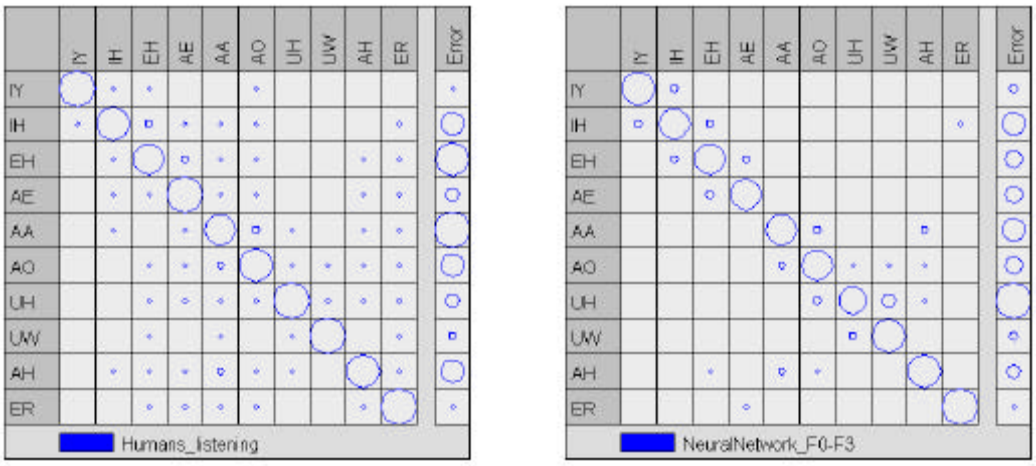



**Figure 6:** Confusion matrices for humans (left) and machine (right). From Klautau 2002.

However, no distinctive AA/AO confusion is noted.

### 4.3.3 Zahorian Vowel Dataset

Zahorian and Jagharghi (1993) set out to explore the effects of feature representation on classification results. Formants were claimed to be more effective than LAR features in the quote above, however neither of these are generally used in ASR. Zahorian compared more traditional ASR cepstral features with the formant representation. The dataset consisted of 11 English vowels.

The baseline human performance was **91**% correct. With fundamental frequency and formants machine performance was **83**%, and with cepstral features, **86**%.

The improvement is not very substantial, but Zahorian points out that the cepstral feature confusion was more similar to human confusion (including the AA/AO trend). This could seem a hidden strong-point of the cepstral features and might support their general success on the ASR task.

In any case the result could suggest that although there is improvement with cepstral features the specific feature representation is not as critical for AVR as for the ASR task.

### 4.3.4 North Texas Vowel Database Dataset

The North Texas Vowel Database contains a large number of vowel samples from 10 men, 10 women and 30 children. Each vowel is repeated around 10 times by each speaker.

The raw audio is available, but it seems that this database has not been used for actual AVR so far. The vowel perception studies, where it has been used, do however provide human confusion matrices.

The average successful recognition rate for human listeners was around **90%** for a subset of three men, and **84%** averaged across three speakers from all speaker groups (Assmann 2001). This suggests that the dataset is difficult compared to the datasets considered earlier.

### 4.3.5 Hillenbrand Vowel Dataset

The Hillenbrand dataset consists of 12 English vowels with 45 male, 48 female and 46 child speakers (Hillenbrand 1995). Each speaker provided one sample per vowel. Hillenbrand's study was originally meant as a replication of the Peterson and Barney study (see above), however considerable differences were found in the recorded vowel data between the two datasets. The original audio is available, and unlike the previous dataset AVR has already been attempted. On a subset of the Hillenbrand vowel dataset consisting of 45 male speakers (and all 12 vowels) the average successful recognition rate was **95%** for human listeners (Hillenbrand 1995). This high recognition rate suggests that the dataset is considerably easier than the North Texas dataset, although 12 vowel samples were misclassified by a majority of listeners. Hillenbrand (2003) reports a **92%** correct recognition rate with narrow band (long window) spectra features and a template-matching classification scheme based on a simple distance measure. This seems to top the list of AVR results, but it should be noted that the templates were created without any samples scoring lower then 85% correct in the human listening condition.

### 4.3.6 Discussion

Generally, we see a wide performance range among the different datasets depending on dataset difficulty and feature representation. A range of different classifiers do quite well, which might suggest that the feature extraction step is more critical. However, the missing 10% or so up to human performance level might be achievable with recurrent approaches that handle the temporal aspects of vowels better. This probably also requires feature vectors different from the pre-computed features in the first three datasets above, so in any case the actual audio data is necessary.

It is also interesting to look for similarities between human and machine performance on the level of vowel confusion trends. Above we saw at least one example, which seems to match fairly well, with the possible exception of the most predominant human confusion trend on AA/AO. We also saw that this confusion is not strongly present in all datasets, and in general some datasets are simply easier than other, also for humans. It appears that good machine performance generally follows human performance with the 10% or so negative off-

set. The very best machine performance is in the range of 7-3% below human performance, with (Hillenbrand 2003) as the most prominent example.

# 5. Automatic Vowel Recognition Results

This section details the experimental method used for AVR with the simple HNN classifier. First the feature extraction and classifier setup are specified, and then we present and discuss the AVR results.

## 5.1 Method

The simple HNN classifier was tested on two datasets of different difficulties.

All tests were performed with unseen speakers. To maximize the utility of the datasets a leave-one-speaker-out paradigm was adopted, i.e. test with one speaker at a time and train with all the rest.

Like traditional AVR and ASR systems the simple HNN classifier can be seen as two separate steps; feature extraction and classification. The feature extraction step entails a number of parameters and the classifier has a single parameter (see respective sections below). The parameters were experimentally adjusted for best results on the individual datasets.

### 5.1.1 Feature Extraction

The primary design criterion for our feature extraction process is that it should be simple enough that it does not exceed the situation in the peripheral auditory system, e.g. it must be without additional (implausible) assistance like principal component analysis or DCT decorrelation.

The three feature extraction techniques considered in the previous section (MFCC, constant-Q and wavelets, the latter two as non-invariant versions) were explored non-exhaustively in an initial informal feature search process. In summary, different settings of MFCC features performed well; however eliminating the DCT step gave considerable improvement. Two different implementations of the constant-Q transform were verified against each other; nevertheless for uninvestigated reasons both failed to provide useful features. Wavelet packet analysis provided results on par with the MFCC features; however wavelets' immense flexibility also means that it was the method tried least exhaustively. We chose to develop upon the MFCC without DCT approach as described below:

The main feature extraction steps correspond to the first two steps of the MFCC procedure described above, i.e. the absolute power spectrum is warped to Mel-scale with triangular overlapping windows, with the difference that our FFT windows are longer than general MFCC windows with the expectation that frequency resolution is more important than time resolution for vowels (Hillenbrand 2003).

The DCT decorrelation step is omitted according to the above criterion and initial results.

Warping to Mel-scale is achieved by a Mel-shaped filter-bank, see figure 7.

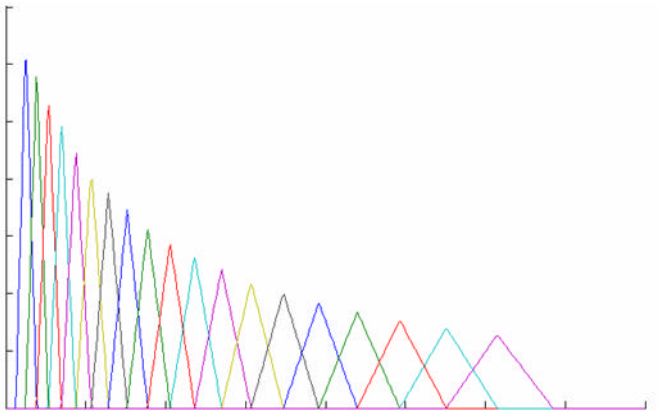


**Figure 7:** Mel-scaled filter-bank with 19 bands.

An overview of the feature extraction process is given in figure 8. The signal is filtered and preemphasized before Fourier transform and warping. The Fourier transform is accelerated by parallel processing on a GPU. Finally we concatenate the Mel-features of each time frame (and truncate the feature vector to 128 elements). This is meant to retain an amount of information about the time development of the vowel spectrum.

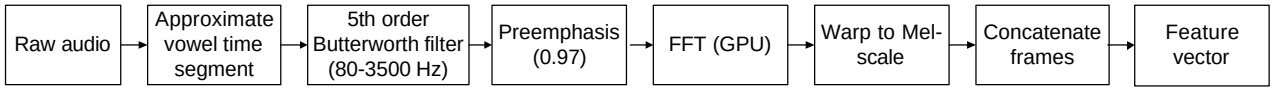



**Figure 8:** Overview of the feature extraction process.

The number of Mel-bands and the window width were experimentally adjusted for each dataset; the remaining parameters were constant.

### 5.1.2 HNN Classification

Classification with the simple HNN classifier proceeds as described in section 3.2. The classifier is implemented in Matlab with certain heavy parts accelerated by GPU though the Matlab MEX interface (see Appendix A). Specifically, convolution and pseudo-inverse de-convolution are accelerated by matrix multiplication on the GPU. The threshold on the diagonal eigenvalues in the de-convolution was experimentally adjusted for each dataset.

## 5.2 Vowel Recognition Results

Automatic vowel recognition was performed on two datasets; the North Texas vowel database and the Hillenbrand vowel dataset. Results are presented below.

### 5.2.1 North Texas Vowel Dataset

Automatic vowel recognition was tested on a subset of the North Texas vowel database consisting of 8 male speakers and 12 vowels; two additional male speakers were disqualified, because data did not exist for all vowels. Each speaker provided a number of speech samples per vowel. All tests were performed with unseen speakers.

MACHINE

| | /i/ | /ɪ/ | /e/ | /ɛ/ | /æ/ | /ʌ/ | /ɜ/ | /ɑ/ | /ɔ/ | /o/ | /ʊ/ | /u/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| /i/ | **85.9** | --- | 2.6 | --- | --- | --- | --- | --- | --- | --- | --- | 11.5 |
| /ɪ/ | 1.5 | **83.8** | 7.4 | 2.9 | --- | --- | --- | --- | --- | 4.4 | --- | --- |
| /e/ | 10.1 | 15.7 | **53.0** | 1.2 | --- | --- | --- | --- | --- | 3.6 | 1.2 | 7.2 |
| /ɛ/ | --- | 30.4 | 1.1 | **60.9** | --- | 7.6 | --- | --- | --- | --- | --- | --- |
| /æ/ | --- | 2.2 | --- | 4.4 | **86.8** | 5.5 | --- | --- | 1.1 | --- | --- | --- |
| /ʌ/ | --- | --- | --- | 2.6 | 2.6 | **80.3** | 2.6 | 1.3 | --- | 2.6 | 7.9 | --- |
| /ɜ/ | --- | 9.7 | --- | --- | --- | 2.2 | **76.3** | --- | --- | --- | 10.8 | 1.1 |
| /ɑ/ | 1.1 | --- | --- | --- | 2.3 | 2.3 | --- | **80.7** | 12.5 | 1.1 | --- | --- |
| /ɔ/ | 1.1 | --- | --- | --- | 6.4 | 2.1 | --- | 42.6 | **44.7** | 3.2 | --- | --- |
| /o/ | --- | --- | --- | --- | --- | --- | 1.2 | --- | --- | **92.9** | 3.5 | 2.4 |
| /ʊ/ | 2.3 | 11.6 | --- | --- | --- | 3.5 | 4.7 | --- | --- | 18.6 | **53.5** | 5.8 |
| /u/ | 3.3 | 2.2 | --- | --- | --- | --- | --- | --- | --- | 3.3 | --- | **91.3** |

HUMAN

| | /i/ | /ɪ/ | /e/ | /ɛ/ | /æ/ | /ʌ/ | /ɜ/ | /ɑ/ | /ɔ/ | /o/ | /ʊ/ | /u/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| /i/ | **97.0** | 2.2 | 0.7 | — | — | — | — | — | — | — | — | — |
| /ɪ/ | — | **65.2** | — | 31.9 | — | — | — | — | — | 0.7 | 2.2 | — |
| /e/ | 0.7 | — | **96.3** | 3.0 | — | — | — | — | — | — | — | — |
| /ɛ/ | — | 4.4 | — | **85.2** | 9.6 | 0.7 | — | — | — | — | — | — |
| /æ/ | — | — | 0.7 | 6.7 | **89.6** | 0.7 | — | 1.5 | 0.7 | — | — | — |
| /ʌ/ | — | — | — | 3.7 | 3.0 | **82.2** | 0.7 | 5.9 | 0.7 | 0.7 | 3.0 | — |
| /ɜ/ | — | — | — | 0.7 | 0.7 | 1.5 | **95.6** | — | — | — | 1.5 | — |
| /ɑ/ | — | — | — | — | — | — | — | **65.2** | 34.8 | — | — | — |
| /ɔ/ | — | — | — | — | — | 0.7 | — | 42.2 | **57.0** | — | — | — |
| /o/ | — | — | — | — | — | — | — | — | — | **100.0** | — | — |
| /ʊ/ | — | — | — | 3.0 | — | 10.4 | 1.5 | — | — | — | **84.4** | 0.7 |
| /u/ | — | 0.7 | — | — | — | — | — | — | — | 2.2 | 6.7 | **90.4** |

**Figure 9:** North Texas confusion matrices: top matrix is HNN result, bottom is listening result from (Assmann 2001).

The average successful recognition rate for human listeners was **84%** across three speakers for each speaker group (Assmann 2001).

The average successful HNN recognition rate (on male speakers) was **74%.** It was unclear which three male speakers were used for Assmann's **90%** listening result; picking the three best speakers from the HNN test resulted in **78%** correct. The highest individual speaker result was 80%, the lowest (and only result below 70%) was 59%; investigation did however not reveal why this single speaker was recognized so poorly.

Figure 9 above shows confusion matrices for our AVR (top) and human listeners averaged over all groups (bottom); ideally we should compare with only males as used in the HNN case, however although listeners' recognition results were different between speaker groups the confusion was generally similar. Interestingly, both matrices show AA/AO confusion (circled in red) similar to the human listeners on the PBvowel dataset. Both human and machine do best on /o/, while the rest of the confusion shows more sporadic similarity.

Parameter settings: Window: 64ms, Overlap: 32ms, Mel bands: 19

Threshold: 0.035

### 5.2.2 Hillenbrand Vowel Dataset

Automatic vowel recognition was tested on a subset of the Hillenbrand vowel dataset consisting of 45 male speakers and 12 vowels. Each speaker provided one speech sample per vowel. All tests were performed with unseen speakers, and a leave-one-speaker-out paradigm.

The average successful HNN recognition rate was **84%**, for 15 speakers classification was perfect. The average successful recognition rate on the same data was **95%** for human listeners (Hillenbrand 1995) and Hillenbrand (2003) reported **92%** for machine classification. 12 vowel samples were misclassified by a majority of listeners. Removing these samples from the dataset resulted in an 85% recognition rate, suggesting that the system can reasonably tolerate an amount of noise in the data. Figure 10 compares HNN and human confusion. The Hillenbrand dataset does not show the typical AA/AO confusion trend very clearly; in fact the human results are so good that it is difficult to compare the confusions. Hillenbrand (2003) notes a limitation of their model evident in a tendency to confuse high front vowels with high back vowels (/i/, /I/ and /u/, /U/); although our result is considerable lower this confusion trend does not seem to be present.

It is also interesting that the optimal threshold in the de-convolution procedure was found to be lower than for the North Texas dataset, which is according to expectation since the Hillenbrand dataset is easier (implying less noise in the data). Parameter settings: Window: 80ms. Overlap: 40ms. Mel bands: 13. Threshold: 0.01

MACHINE

| | /i/ | /ɪ/ | /e/ | /ɛ/ | /æ/ | /ʌ/ | /ɜ/ | /ɑ/ | /ɔ/ | /o/ | /ʊ/ | /u/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| /i/ | **84.4** | 2.2 | 2.2 | 11.1 | --- | --- | --- | --- | --- | --- | --- | --- |
| /ɪ/ | --- | **80.0** | 8.9 | --- | --- | 11.1 | --- | --- | --- | --- | --- | --- |
| /e/ | --- | 6.7 | **88.9** | --- | --- | 4.4 | --- | --- | --- | --- | --- | --- |
| /ɛ/ | 15.6 | --- | 2.2 | **80.0** | --- | --- | --- | --- | 2.2 | --- | --- | --- |
| /æ/ | --- | --- | --- | 2.2 | **88.9** | --- | --- | --- | 4.4 | 4.4 | --- | --- |
| /ʌ/ | 2.2 | 11.1 | 6.7 | --- | --- | **66.7** | --- | --- | --- | --- | --- | 13.3 |
| /ɜ/ | --- | --- | --- | --- | --- | --- | **91.1** | --- | --- | --- | --- | 8.9 |
| /ɑ/ | --- | --- | --- | --- | --- | --- | --- | **86.7** | --- | --- | --- | 13.3 |
| /ɔ/ | --- | --- | --- | --- | 4.4 | --- | --- | 2.2 | **84.4** | 2.2 | --- | 6.7 |
| /o/ | --- | --- | --- | --- | 6.7 | --- | --- | --- | 2.2 | **91.1** | --- | --- |
| /ʊ/ | --- | --- | --- | --- | --- | --- | 2.2 | 2.2 | --- | --- | **82.2** | 13.3 |
| /u/ | --- | --- | --- | --- | 2.2 | --- | --- | 8.9 | --- | --- | --- | **88.9** |

HUMAN

| | /i/ | /ɪ/ | /e/ | /ɛ/ | /æ/ | /ʌ/ | /ɜ/ | /ɑ/ | /ɔ/ | /o/ | /ʊ/ | /u/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| /i/ | **99.9** | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| /ɪ/ | --- | **99.4** | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| /e/ | --- | --- | **99.8** | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| /ɛ/ | --- | 2.2 | --- | **86.1** | 11.1 | --- | --- | --- | --- | --- | --- | --- |
| /æ/ | --- | --- | --- | 6.9 | **92.2** | --- | --- | 0.6 | --- | --- | --- | --- |
| /ʌ/ | --- | --- | --- | --- | --- | **88.0** | --- | 6.4 | 4.7 | --- | 0.7 | --- |
| /ɜ/ | --- | --- | --- | --- | --- | --- | **99.6** | --- | --- | --- | --- | --- |
| /ɑ/ | --- | --- | --- | --- | --- | 2.5 | --- | **86.0** | 11.3 | --- | --- | --- |
| /ɔ/ | --- | --- | --- | --- | --- | 3.6 | --- | 3.5 | **92.6** | --- | --- | --- |
| /o/ | --- | --- | --- | --- | --- | --- | --- | --- | --- | **98.8** | --- | 1.0 |
| /ʊ/ | --- | --- | --- | --- | --- | 1.9 | --- | --- | --- | --- | **96.1** | 1.3 |
| /u/ | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | 1.7 | **98.0** |

**Figure 10:** Hillenbrand dataset confusion matrices for males. Top is HNN result, bottom human.

### 5.2.3 Discussion

It is difficult to evaluate the performance of the simple HNN classifier without a direct comparison to other approaches on the same datasets and features; however we have seen that a general trend between different datasets is that good machine performance is about 10% lower than human performance. This is a reasonable clue in the evaluation and suggests that the simple HNN classifier performs quite well, and it is good news considering its relative simplicity and distributed plausibility. Whether this scales to the ASR-task is uncertain, but it is clear from the results that the simple HNN classifier can interface with features similar to typical ASR features, without the noise-like mapping of earlier HNN.

It is also clear that the very best of earlier AVR approaches outperform our HNN; however this is not a major concern, since we did not set out to break the record, but rather to demonstrate that our improvement to the HNN paradigm function as expected. It is also reasonable that better recognition rates can be achieved with more carefully constructed features.

# 6. Conclusion

We have implemented a holographic neural network classifier with distributed representation based on a holographic analogy to circular convolution and de-convolution.
We replaced the circular correlation approximation used for de-convolution in earlier holographic neural networks with the Moore-Penrose pseudo-inverse thereby removing a previous restriction to random (noise-like) feature vectors. Our implementation of the pseudo-inverse supports regularization similar to truncated singular value decomposition, which we use for noise reduction in the classifier. The computation of circular convolution and de-convolution is accelerated by parallel processing on a GPU and we use the real-valued Hartley transform instead of Fourier for further speed-up.

The holographic neural network classifier performs on par with most other good automatic vowel recognition approaches although a direct comparison with other classifiers has not been done. On one dataset we achieved 74% correct classification versus 84% for humans; on another dataset we achieved 84% versus 95% for humans.
The automatic vowel recognition results constitute a reasonable success; importantly, we note that a holographic neural network, using the new pseudo-inverse de-convolution procedure, can interface with simple spectral features (without the need for special mapping to noise-like features).

# 7. Acknowledgements

Special thanks to Dr. Ronald van Elburg. Many thanks to Dr. Tjeerd Andringa, Maria Niessen, Dirkjan Krijnders and the rest of the Auditory Cognition Group at RUG.

Thanks to my family and friends.

# 8. References


**Anderson 1983:** John R. Anderson, A *spreading activation theory of memory*, Journal of Verbal Learning and Verbal Behavior, 1983.

**Anderson 1993:** Anderson, J. R., *Rules of the mind*. Hillsdale, NJ: Erlbaum.

**Anderson and Lebiere 1998:** John R. Anderson, Christian V. Lebiere, *The Atomic Components of Thought*, Lawrence Erlbaum Associates Inc.

**Assmann 2001:** William F. Katz and Peter F. Assmann, *Identi5cation of children's and adults' vowels: intrinsic fundamental frequency, fundamental frequency dynamics, and presence of voicing*, *Journal of Phonetics* (2001) 29, 23}51

**Astakhov 2007:** Vadim Astakhov, Tamara Astakhova, Brian Sanders, *Imagination as Holographic Processor for Text Animation*, SIGCHI 2007, arXiv:cs/0606020

**Blankertz 1999:** Benjamin Blankertz, *A Robust Constant Q Spectrum for Polyphonic Pitch Tracking*, CCRMA DSP Seminar, Thursday June 3, 3:15pm, CCRMA Ballroom, 1999

**Blankertz 199?:** Benjamin Blankertz, *The Constant Q Transform*, date unknown, 199?

**Brown 1991:** Brown, J.C., *Calculation of a Constant Q Spectral Transform,* J. Acoust. Soc. Am. 89, 425-434.

**Brown and Puckette 1992:** Brown, J.C. and Puckette, M.S. *An Efficient Algorithm for the Calculation of a Constant Q Transform,* J. Acoust. Soc. Am. 92, 2698-2701.

**Brown 2008:** Brown, J.C, 2008, http://web.media.mit.edu/~brown/cqtrans.htm

**Chatterji 2003:** Shourov K. Chatterji, *The fast discrete Q-transform*, LIGO Scientific Collaboration Meeting November 10-13, 2003

**Cheng 1997:** Lizhi Cheng, *Fast Hartley transform and truncated singular value algorithm for circular deconvolution*, Opt. Eng Vol. 36, 2137, 1997

**Davis 1998:** Geoff Davis, *A Wavelet-based Analysis of Fractal Image Compression*, *IEEE Transactions on Image Processing*, Feb. 1998.

**Dennett 1991:** Daniel C. Dennett, *Real Patterns*, The Journal of Philosophy 88, 1991, 27–51.

**Deterding 1989:** D. H. Deterding, *Speaker Normalisation for Automatic Speech Recognition*, University of Cambridge, submitted for PhD.

**Deviren 2003:** Murat Deviren, Khalid Daoudi, *Frequency Filtering or Wavelet Filtering?*, Joint 13th Int. Conf. on Artificial Neural Networks and 10th Int. Conf. on Neural Information Processing (ICANN/ICONIP) June 26-29 2003, Istanbul, Turkey

**De Vito 2005:** De Vito E., Rosasco L., Caponnetto A., De Giovannini U., Odone F., *Learning from Examples as an Inverse Problem*, Journal of Machine Learning Research, 6 883-904 (2005).

**Doctorow 2001:** Cory Doctorow, *Metacrap: Putting the torch to seven straw-men of the meta-utopia*, Version 1.3: 26 August 2001.

**Eliasmith 1997:** Eliasmith, C., *Structure without symbols: Providing a distributed account of high-level cognition,* Southern Society for Philosophy and Psychology. Conference March, 1997.

**Farmelo 2002:** Graham Farmelo (ed), *It Must be Beautiful: Great Equations of Modern Science*, Granta Books

**Fernando 2003:** *Pattern Recognition in a Bucket*, C. Fernando, S. Sojakka, ECAL 2003, 2003-09

**Fodor and Pylyshyn 1988:** Fodor, J. A., and Z. Pylyshyn. *Connectionism and cognitive architecture: A critical analysis*. Cognition 28: 3–71.

**Fodor and McLaughlin 1990:** Fodor, J. and B. McLaughlin, *Connectionism and the problem of systematicity: Why Smolensky's solution doesn't work.* Cognition 35: 183-204.

**Ganapathiraju 1998:** A. Ganapathiraju, J. Hamaker & J. Picone, *Support vector machines for speech recognition*, ICSLP'98.

**Gelder and Niklasson 1994a:** van Gelder, T. J., & Niklasson, L. *On being systematically connectionist.* Mind and Language, 9, 288-302.

**Gelder and Niklasson 1994b:** van Gelder, T., & Niklasson, L. *Classicalism and cognitive architecture.* In Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society. Hillsdale NJ: Erlbaum

**Grossberg 2003:** Grossberg, S. & Carpenter, G.A., *Adaptive Resonance Theory*, In M.A. Arbib (Ed.), The Handbook of Brain Theory and Neural Networks, Second Edition (pp. 87-90). Cambridge, MA: MIT Press

**Hansen 1987:** Per Christian Hansen, *The truncated SVD as a method for regularization*, BIT, 27 (1987), pp. 534-553.

**Hansen 1996:** Hansen, P. C. and Jensen, S. H., *Filter Model of Reduced-Rank Noise Reduction.* In Proceedings of the Third international Workshop on Applied Parallel Computing, industrial Computation and Optimization (August 18 - 21, 1996). J. Wasniewski, J. Dongarra, K. Madsen, and D. Olesen, Eds. Lecture Notes In Computer Science, vol. 1184. Springer-Verlag, London, 379-387.

**Haw 2003:** Haw M., *Holographic data storage: The light fantastic*, Nature 2003 Apr. 10;422(6932):556-8.

**Herman 2007:** Joshua Jay Herman, *Liquid State Machines in Adbiatic Quantum Computers for General Computation*, Submitted on 6 Sep 2007, arXiv:0709.0883v3

**Hillenbrand 1995:** James Hillenbrand, Laura A . Getty, Michael J. Clark , and Kimberlee Wheeler, *Acoustic characteristics of American English vowels*, J Acoust Soc Am. 1995 May;97(5 Pt 1):3099-111.

**Hillenbrand 2003:** James M. Hillenbrand, Robert A Houde, *A narrow band pattern-matching model of vowel perception*, The Journal of the Acoustical Society of America, Vol. 113, No. 2. (2003), pp. 1044-1055.

**Hinton 1986:** Hinton GE, *Learning distributed representations of concept.*, In Proceedings of the Eighth Annual Conference of the Cognitive Science, 1-12 Hillsdale, NJ: Erlbaum. (Reprinted in Parallel Distributed Processing: Implications for Psychology and Neurobiology, Ed. Morris RGM (1989), pp46-61, Oxford University Press.

**Hinton 1990:** Hinton, G. E. 1990. *Mapping part-whole hierarchies into connectionist networks.* Artif. Intell. 46, 1-2 (Nov. 1990), 47-75.

**Jaeger 2001:** Jaeger H., *The "echo state" approach to analysing and training recurrent neural networks.* GMD Report 148, GMD - German National Research Institute for Computer Science

**Kanerva 1998:** Kanerva, P., *Large patterns make great symbols: An example of learning from example.* In Neural Information Processing Systems Conference 1998 (NIPS98), Denver, Colorado, U.S.A.

**Klautau 2002:** Aldebaro Klautau, *Classification of Peterson & Barney's vowels using Weka*, Technical report, UFPA, 2002.

**Klautau 2008:** Aldebaro Barreto da Rocha Klautau Júnior, *Vowel (Deterding) database* http://www.laps.ufpa.br/~aldebaro/repository/vowel.htm, 2008

**Kudrolli 1998:** A. Kudrolli, B. Pier, and J.P. Gollub, *Superlattice Patterns in Surface Wave*, Physica D, Volume 123, Number 1, 15 November 1998 , pp. 99-111(13)

**Kurzweil 2005:** Kurzweil, Ray *(2005), The Singularity is Near, Penguin Books, ISBN 0-670-03384-7*

**Kvasnicka 2006:** Vladimír Kvasnicka, Jirí Pospíchal, *Deductive rules in holographic reduced representation*, Neurocomputing, Volume 69, Issues 16-18, October 2006, Pages 2127-2139

**Lebiere and Anderson 1993:** Lebiere, C. & Anderson, J. R., *A connectionist implementation of the ACT-R production system*. In: Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society, (pp. 635–640). Hillsdale, NJ: Erlbaum.

**Lehar 2004:** Steven Lehar, *Harmonic Resonance Theory: an Alternative to the "Neuron Doctrine" Paradigm of Neurocomputation to Address Gestalt properties of perception*, Submitted to Journal of Integrative Neuroscience August 2004.

**Leith 1964:** E. N. Leith and J. Upatnieks, *Wavefront Reconstruction with Diffused Illumination and Three-Dimensional Objects*, J. Opt. Soc. Am. **54**, 1295- (1964)

**Levy 2006:** Levy, S.D. *Distributed Representations,* Philosophy 395, Presentation.

**Levy 2007:** Levy, S.D. (2007) *Changing Semantic Role Representations with Holographic Memory*. In Computational Approaches to Representation Change during Learning and Development: Papers from the 2007 AAAI Symposium. Technical Report FS-07-04, AAAI Press.

**Lieberman 2005:** Lieberman Henry, Faaborg Alexander, Daher Waseem, Espinosa José, *How to wreck a nice beach you sing calm incense*, Proceedings of the 10th international conference on Intelligent user interfaces, January 10-13, 2005, San Diego, California, USA

**Linzer 1992:** Elliot Linzer, *On the Stability of Transform-Based Circular Deconvolution*, SIAM Journal on Numerical Analysis, Vol. 29, No. 5. (Oct., 1992), pp. 1482-1492.

**Long 1996:** C.J. Long, S. Datta, *Wavelet based feature extraction for phoneme recognition*, Proceedings of the 4th International Conference of Spoken Language Processing Philadelphia, USA Oct.1996, Vol. 1, pp. 264–267.

**Maass 2002:** W. Maass, T. Natschläger, and H. Markram. *Real-time computing without stable states: A new framework for neural computation based on perturbations*. Neural Computation, 14(11):2531-2560, 2002.

**Marshall and Shipman 2003:** Catherine C. Marshall , Frank M. Shipman, *Which semantic web?*, Catherine C. Marshall , Frank M. Shipman, Proceedings of the fourteenth ACM conference on Hypertext and Hypermedia.

**McCarthy 2007:** John McCarthy, *What is Artificial Intelligence*, http://www-formal.stanford.edu/jmc/whatisai.html, 2007

**Mitchell 1997:** Tom Mitchell, *Machine Learning*, textbook, McGraw Hill, 1997.

**Mporas 2007:** Iosif Mporas, Todor Ganchev, Mihalis Siafarikas, Nikos Fakotakis, *Comparison of Speech Features on the Speech Recognition Task*, Journal of Computer Science 3 (8): 608-616, 2007

**Mussa-lvaldi 2007:** Ferdinando A. Mussa-lvaldi, Lee E. Miller, W. Zev Rymer, and Richard, *Neural Engineering*, in Parasuraman R., Rizzo M. (Eds.) Neuroergonomics: The Brain at Work, p.306, Oxford University Press 2007

**Neumann 2001:** Neumann, Jane. *Holistic Processing of Hierarchical Structures in Connectionist Networks.* PhD Thesis, University of Edinburgh, Scotland, August 2001.

**Niklasson 1994:** Niklasson, L.F. and van Gelder, T. J, *On being systematically connectionist*, Mind and Language, 9(3), 289.302.

**Niklasson and van Gelder 1994:** Niklasson, L. and van Gelder, T. *Can connectionist models exhibit non-classical structure sensitivity?* In Proceedings of the Sixteenth Annual Conference of the Cognitive Society 1994, Atlanta, pages 664 – 669, Hillsdale, NJ. Lawrence Erlbaum Associates.

**Nvidia 2006:** *CUDA: A New Architecture for Computing on the GPU*, Nvidia, 2006

**Nvidia 2007:** *Accelerating MATLAB with CUDA Using MEX Files*, Nvidia, WP-03495-001_v01, September 11, 2007

**Passin 2004:** Thomas B. Passin, *Explorer's Guide to the Semantic Web*, Manning Publications Co. Published, June, 2004.

**Penrose 1989:** Roger Penrose, *The Emperor's New Mind*, Oxford University Press, Oxford, *New* York, Melbourne, 1989

**Penrose 2005:** Roger Penrose, *The Road to Reality: A Complete Guide to the Laws of the Universe*, Vintage Books, 2005, ISBN 0-09-944068-7

**Plate 1991:** Plate, Tony. *Holographic Reduced Representations: Convolution Algebra for Compositional Distributed Representations*, In P. Mehra and B.W. Wah, editors. Artificial neural networks: concepts and theory. IEEE Computer Society Press Tutorial, 1992, 6 pages. (Note: this is a reprint of IJCAI91 paper.)

**Plate 1993:** Plate, Tony. *Holographic recurrent networks*, In C. L. Giles, S.J. Hanson, and J. D. Cowan, editors, Advances in Neural Information Processing Systems 5 (NIPS*92), pp34-41, Morgan Kaufmann, San Mateo, CA, 1993.

**Plate 1995:** Tony A. Plate, *Holographic Reduced Representations*, IEEE Transactions on Neural Networks, vol. 6, no. 3, pp623-641, 1995.

**Poggel 2007:** Dorothe A. Poggel, Lotfi B. Merabet and Joseph F. Rizzo, *Artificial Vision*, in Parasuraman R., Rizzo M. (Eds.) Neuroergonomics: The Brain at Work, p.337, Oxford University Press 2007

**Pollack 1990:** Pollack, J. B. *Recursive distributed representations.* Artificial Intelligence, 46:77–105.

**Rabal 2001:** Rabal H., *Holographic analogues*, Optik – International Journal for Light and Electron Optics, Volume 112, Number 4, April 2001 , pp. 153-156(4)

**Robinson 1989:** A. J. Robinson, *Dynamic Error Propagation Networks*, Cambridge University Engineering Department.

**Rosasco 2004:** Rosasco L., Caponnetto A., De Vito E., De Giovannini U., Odone F., *Learning, Regularization and Ill-Posed Inverse Problems*, Conference proceedings: Eighteenth Annual Conference on Neural Information Processing Systems, 2004.

**Smolensky 1990:** Smolensky, P., *Tensor product variable binding and the representation of symbolic structures in connectionist systems*. Artificial Intelligence, 46:159–216.

**Sarikaya 1998:** R. Sarikaya, B. L. Pellom, and J. H. Hansen, *Wavelet Packet Transform Features with Application to Speaker Identification*, NORSIG'98, pp. 81--84, 1998

**Schiller 2005:** Ulf D. Schiller and Jochen J. Steil, *Analyzing the weight dynamics of recurrent learning algorithms, Neurocomputing*, volume 63, January 2005, Pages 5-23

**Schönemann 1987:** Schönemann PH, *Some algebraic relations between involutions, convolutions, and correlations, with applications to holographic memories*, Biol Cybern. 1987;56(5-6):367-74. Erratum in: Biol Cybern. 2008 Apr;98(4):355.

**Searle 1996:** John Searle, *Philosophy of Mind - Lectures by John Searle*, The Teaching Company, 1996

**Shapiro 1992:** Stuart C. Shapiro, *Artificial Intelligence*. In S. C. Shapiro, Ed. Encyclopedia of Artificial Intelligence, Second Edition. John Wiley & Sons, Inc., New York, 1992, 54-57

**Shirky 2003:** Clay Shirky, *The Semantic Web, Syllogism, and Worldview*, November 7, 2003 on the "Networks, Economics, and Culture" mailing list.

**Siafarikas 2004:** T. Ganchev, M. Siafarikas, N. Fakotakis: *Speaker Verification Based on Wavelet Packets* , 7th International Conference on Text Speech and Dialogue (TSD 2004), Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, ISSN: 0302-9743, vol. LNAI 3206/2004, Sept. 2004, pp. 299–306.

**Siafarikas 2005:** M. Siafarikas, T. Ganchev, N. Fakotakis, G. Kokkinakis: *Overlapping Wavelet Packet Features for Speaker Verification* , INTERSPEECH 2005, September 4-8, 2005. Lisbon, Portugal, pp.3121-3124.

**Smolensky 1990:** Smolensky, P., *Tensor product variable binding and the representation of symbolic structures in connectionist systems*, Artificial Inteligence. 46, 1-2 (Nov. 1990), 159-216.

**Thagard 2001:** Eliasmith, C., Thagard P., *Integrating Structure and Meaning: A Distributed Model of Analogical Mapping.* Cognitive Science.", Cognitive Science, Volume 25, Number 2, March 2001 , pp. 245-286(42)

**Thagard 2005:** Paul Thagard, *Mind: Introduction to Cognitive Science*, second edition, MIT Press, 2005.

**Westlake 1970:** Westlake, PR., *The Possibilities of Neural Holographic Processes within the Brain*, Kybernetik. 1970 Sep;7(4):129-53.

**Wilson 2002:** Margaret Wilson, *Six views of embodied cognition*, Psychonomic Bulletin & Review, Volume 9, Number 4, 1 December 2002 , pp. 625-636(12)

**Zahorian and Jagharghi 1993:** Zahorian SA, Jagharghi AJ., *Spectral-shape features versus formants as acoustic correlates for vowels*, J Acoust Soc Am. 1993 Oct;94(4):1966-82.

# 9. Appendix A: Introduction to GPU with perspectives to HNN

This section is a short introduction to parallel processing and the parallel graphics processing unit (GPU). We will also consider some perspectives to holographic neural networks (HNN) and the simple HNN classifier.

When it comes to actually executing the algorithms of the simple HNN classifier or other AI systems we need processing power enough that training and classification proceeds in reasonable time. It is natural to consider parallel processing in this context, since the brain is a highly parallel system, but parallel processing is generally more difficult to implement and it is not always straight forward to work out when to apply it to an advantage.

There is no fundamental difference between serial processing and parallel processing in the sense that a serial process can simulate parallel processing. However, some problems can be treated more efficiently in parallel. The historically most prominent example of this is computer graphics (CG); hence parallel processors are commonly called GPUs. More general scientific processing has long been quite difficult to implement, because the hardware and development language was specifically focused on CG. In a sense other scientific problems have had to be translated into CG problems; this has basically been the approach of most general purpose computing on GPU (GPGPU), e.g. Stanford's BrookGPU framework.

In the meantime mainstream GPU tools have been moving towards easier CG development with for instance Microsoft's high-level shader language (HLSL) and other tools of high what-you-see-is-what-you-get abstraction. However, specific hardware and software for GPGPU have only recently become available. The GPU manufacturer Nvidia has released a GPGPU package called CUDA and the GPU rival ATI also has a similar product. CUDA offers GPGPU improvements at the hardware and driver level, as well as high-level libraries for basic linear algebra (BLAS) and various fast Fourier transforms (FFT). This makes it possible to easily call GPU functionality from C and C++ code hosted on a (serial) central processing unit (CPU). This process is referred to as "on-loading" to GPU. Figure 11 shows a C++ code example of on-loading matrix multiplication to GPU.

```
/* Allocate device memory for the matrices */
status = cublasAlloc(m, sizeof(float), (void**)&d_A);
status = cublasAlloc(mA*mB, sizeof(float), (void**)&d_B);
status = cublasAlloc(mA*mB, sizeof(float), (void**)&d_C);

/* Initialize the device matrices with the host matrices */
status = cublasSetVector(m, sizeof(float), A, 1, d_A, 1);
status = cublasSetVector(mA*mB, sizeof(float), B, 1, d_B, 1);

/* Performs operation using cublas */
cublasSgemm('n', 'n', mA, mB, mA, alpha, d_A, mA, d_B, mA, beta, d_C, mA);

/* Read the result back */
status = cublasGetVector(mA*mB, sizeof(float), d_C, 1, C, 1);
```

**Figure 11:** On-loading matrix multiplication to GPU.

Another valuable aspect of CUDA is that GPU on-loading can also be done from Matlab through the MEX interface (see Nvidia 2007 for details).

Matrix multiplication and FFT represent most of the computational complexity of the simple HNN classifier and both problems intuitively parallelize well with a divide-and-conquer strategy. In general many scientific problems lend themselves to effective parallelization; figure 12 gives some estimates (probably to be taken with a grain of salt) by Nvidia (Nvidia 2006):

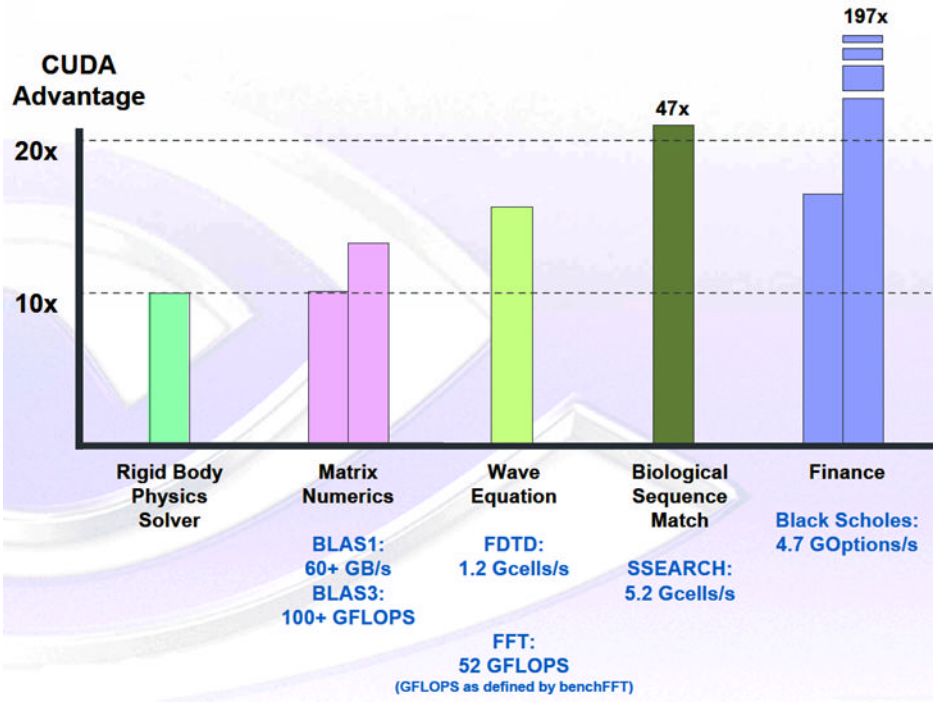



**Figure 12:** CUDA speed-up on various problem types.

In relation to the future directions for HNN considered earlier it is interesting to note that liquid state machines (LSM) have been implemented with a real bucket of water as the liquid medium (Fernando 2003) and that simulating water surfaces in CG is traditionally a strong-point of the GPU, i.e. it might be feasible to simulate a liquid surface on the GPU to implement a similar variation of LSM. As an example of liquid surface simulation figure 13 is a screenshot of the front-page artwork seen through a GPU simulated liquid surface being perturbed by (radial) point-sources along the red "diagonal" (red dots).

It should also be noted that the principles of holography can be applied to a simulated liquid medium. The wave pattern created by a single point-source (concentric circles of outward decreasing thickness) is essentially a holographic Fresnel lens. The wave interference pattern created by multiple point-sources can encode more complex information, the photo in figure 14 shows a very simple holographic reconstruction of the letters "A B" done by shining a laser-pen through a transparent print-out of a wave interference pattern; the pattern was created by (mathematically) propagating point-source waves from each black pixel of

"A B" (the inside of the letters, the print). Some patterns might be particularly interesting; depending on the characteristics of the liquid medium and of the reservoir specific sets of patterns are formed at certain resonant harmonic frequencies. These "Chladni" patterns represent *normal modes* (see e.g. Penrose 2005) of the dynamic system and any motion (waves) in the system can be expressed as a superposition of these modes (standing wave patterns). This property has been claimed to support invariant pattern recognition among other relevant cognitive features by Lehar (2004). Rabal (2001) even describes the principles of Chladni patterns as a valid analogy to holography.

Figure 15 shows a quasi-crystal (Penrose tiling) normal mode pattern in a liquid (Kudrolli 1998); in an equivalent simulation much greater control over both the medium and reservoir would be expected (the simulation behind figure 13 is a simple example where wave propagation speed and frequency can be adjusted during simulation), and with greater control even more complex patterns can be imagined.

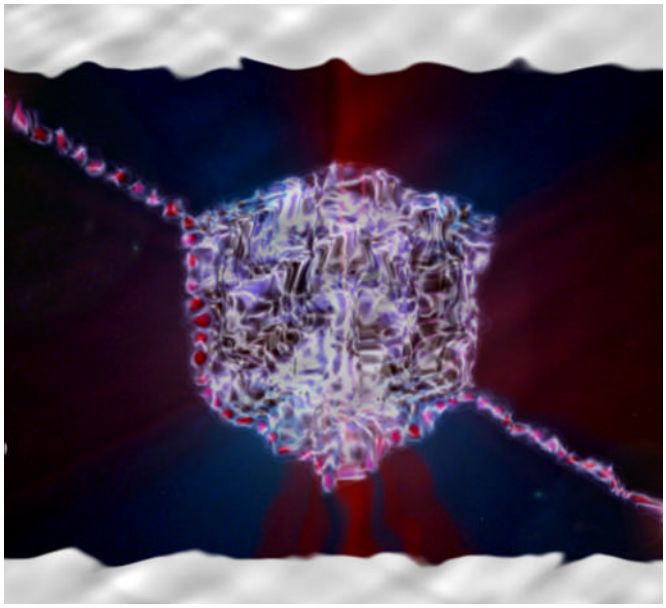


**Figure 13:** Screenshot of liquid surface GPU simulation

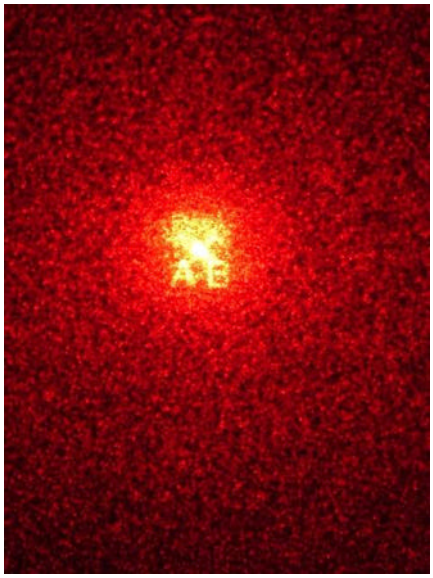


**Figure 14**: Primitive holographic reconstruction

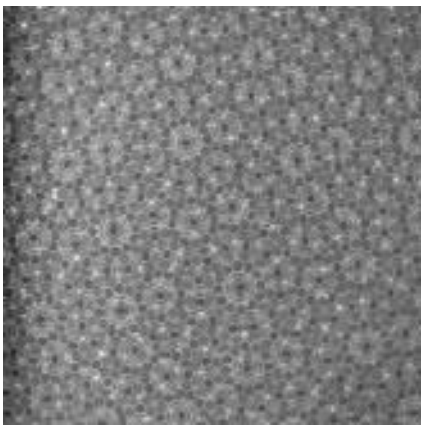


**Figure 15:** Quasi-crystal pattern in liquid. From Kudrolli 1998.

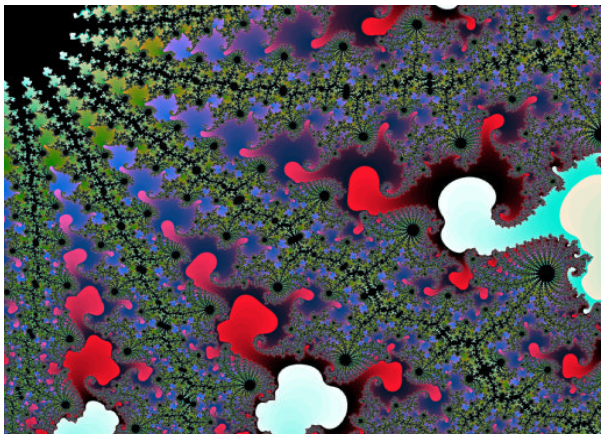


**Figure 16:** Interactive GPU based Mandelbrot set

This brings us to the final highlight of problems that parallelize well onto GPU: Figure 16 show a screenshot of an interactive representation of the Mandelbrot set done with CUDA. The summary conclusion is that fractal's iterative nature also fit the GPU well.
These examples point to the continued relevance of the GPU in relation to the future directions of HNN research suggested in previous sections.