מכון ויצמן למדע

WEIZMANN INSTITUTE OF SCIENCE

Thesis for the degree
Master of Science

חבור לשם קבלת התואר
מוסמך למדעים

By
Uri Klein

מאת
אורי קליין

מודל רשת נוירוביולוגי לבחירת היררכיות מאפיינים עבור זיהוי עצמים

A Neurobiological Feature Hierarchies Extraction Network Model for Object Classification

Advisor
Prof. Shimon Ullman

מנחה
פרופ' שמעון אולמן

April, 2006

ניסן, תשס"ו

Submitted to the Scientific Council of the
Weizmann Institute of Science
Rehovot, Israel

מוגש למועצה המדעית של
מכון ויצמן למדע
רחובות, ישראל

I Would Like to Thank

Professor Shimon Ullman, my supervisor in this research, for accepting me as his student, for showing me what research work is all about and for his great help all along

Sonia Bachor, for her valuable professional help and support

Keren, my wife, who supported me throughout my studies and research

Amit, who contributed his time, mind and computer for me

Abstract

Classification usually proceeds in two main stages: feature detection and decision making. A crucial part of successful classification is extracting useful features during a learning stage. Physiological experiments have shown that biological visual systems extract feature hierarchies. Moreover, recent computational work suggests that class-specific, hierarchical features are more useful and informative than undivided, integral (generic) features. Previous biological models have suggested the use of hierarchical features, but no model has been developed (to date) that can extract in a biologically plausible manner a hierarchy of useful features for classification. This work presents an artificial neural network model that is able to extract a useful, class-specific, hierarchical set of features in a neurobiologically plausible manner that is inspired by several known aspects of cortical behavior. The simulated learning scheme combines key features from existing classification models such as one-shot learning via neuronal imprinting and local neuronal weight update rule, together with novel feature selection methods that rank neurons based on their usefulness, to cope with feature redundancy. The work examines the usefulness of the resulting image fragments hierarchies to classification, and compares several biological variations with a computational (non-biological) model.

Contents

1 Abbreviations List

- **ANN(s)** – Artificial Neural Network(s)
- **AP(s)** – Action Potential(s)
- **BPN(s)** – Back-Propagation Network(s)
- **CED** – Canny Edge Detector
- **CV** – Classification Value
- **EER(s)** – Equal Error Rate(s)
- **FA(s)** – False Alarm(s)
- **FBC** – Fragment Based Classification
- **FGDM** – Full Gradient-Descent Model
- **HGDM** – Half Gradient-Descent Model
- **IC** – Intermediate Complexity
- **IT** – Infero-Temporal
- **LGB** – Lateral Geniculate Body
- **MFM** – Multiple Feature Model
- **MI** – Mutual Information
- **MV** – Merit Value
- **NGDM** – No Gradient-Descent Model
- **OGDM** – Only Gradient-Descent Model
- **ROC** – Receiver Operating Characteristic
- **ROI** – Region Of Interest
- **RV** – Redundancy Value

2 Introduction

Object-classification is the decision problem of identifying the existence or non-existence of a specific class of objects in an image (e.g. Airplanes, Cows, Men, etc.). The goal of a classification model or algorithm is to construct a machine that can be trained to make this decision for a selected class with as few mistakes as possible. The mistakes include 'False Alarms' (FAs) and 'misses' – a FA happens when declaring 'class' on a non-class image, and a miss happens in the opposite case. A key component of a successful classification scheme is the choice of the features used for reaching a classification decision. In its essence, any classification scheme is constructed of some bank of features, which are searched and examined in any given image. The results of evaluating these features in a given image are finally processed together, to reach a classification conclusion. One particular type of features, image patches or fragments taken from the training images, has been used in several recent classification models to achieve very good results [1, 2, 3, 4, 5].

2.1 Classification by a Fragment Hierarchy

It is reasonable to expect that searching known class patches in a given image should convey information on the appearance of a class object in the new image. It is not clear, however, which fragments should be used, and with which characteristics. In the huge space of possible fragments, the designer of a classification model must know on which type of fragments to focus: large or small? High resolution or low resolution fragments? There is a fundamental tradeoff between the ability of a fragment to fit many class containing images and its non-existence in non-class containing images. The complexity of a fragment (which is higher for large or high-resolution fragments) is linked with the probability of detecting it in an image: a class object fragment which is very specific will be found in very few class images while a very general fragment will be found in many images – regardless of their class relevance. In [6], Ullman et al. showed that Intermediate Complexity (IC) features are optimal for the classification task. In this work, Ullman et al. also showed that a well chosen set of IC fragments, taken from a training-set of some class, may be an optimal set for classification of that class (FBC – Fragment Based Classification).

Once image fragments were used as selected features for the classification task, it was natural to look for a hierarchical development of this idea (as in [7]). The underlying assumption is that any class-object's fragment may be considered as a new object that has to be identified. In an identical manner to the way a class object can be recognized using a set of image fragments, a class-object fragment can be recognized using a set of its own sub-fragments. This hierarchical decomposition of the object-classification problem improves the FBC method by improving the fragments detection. Such a hierarchical set of fragments and sub-fragments can be extended to additional levels that will improve the set's classification ability, until reaching the level of small enough 'atomic fragments' that cannot be efficiently decomposed. A well-constructed hierarchy of class image fragments, can be used bottom-up to recognize the various fragments and finally to classify an image.

2.2 Biological Modeling

The human brain also 'implements' a remarkably fast and flexible classification algorithm in the visual system. Already at a young age a child can distinguish objects such as a cow from a cat and recognize his parents among other adults. Although many computational classification models use complex mathematical calculations and algorithms that are not necessarily feasible in a neural network as the human brain, no model has yet come close to the human visual classification abilities.

This fact gives some motivation to the development of semi-biological classification models that will try to imitate the mechanism of action of the visual system.

The FBC method and its hierarchical development have some properties in common with the biological system. Therefore, adapting this scheme to a semi-biological model may give us some information on possible mechanisms in the brain. There is evidence that the human visual system has some hierarchical structure of increasing complexity. It is also known today that it includes neurons that react to the presentation of some abstract shapes or patterns that may be considered as fragments. One computational model that can be exploited for the attempt to adapt the hierarchical FBC method to a semi-biological system is the Artificial Neural Network (ANN) model.

Several attempts were made in the past to construct ANN models for the purpose of object classification [8, 9, 10, 11, 12]. In many of them, however, the motivation was more to explore the abilities of the ANN model rather than to try and suggest a biologically plausible model. This work presents an ANN classification model that is inspired by the FBC model and by its hierarchical extension, which tries to keep its biological plausibility by using simple local calculations in the network's neurons. The model suggested here performs an on-line learning algorithm of feature selection that constructs a hierarchical set of class image fragments. The informative set of fragments is selected from a larger, exhaustive set of fragments, which is extracted from class images. The algorithm described in this work tries to maximize the information on the class delivered by the set of fragments. The fragments considered as features at the higher levels are of intermediate size and therefore match the conclusions from the FBC method.

3 Background

3.1 Biological Background

This section will briefly present a number of biological findings and concepts that served as guidelines in the design of the proposed model. Although the model does not perfectly simulate a possible biological system, it does obey these fundamental principles and demonstrates their effectiveness in object classification tasks.

3.1.1 Hierarchical Structure of the Visual System

Already in 1962, Hubel and Wiesel suggested a hierarchical model of the visual neural system based on their experiments in the cat's cortex [13, 14]. According to the hierarchy model by Hubel and Wiesel, the neural network in the visual cortex has a hierarchical structure: LGB (lateral geniculate body) → simple cells → complex cells → lower order hypercomplex cells → higher order hypercomplex cells. They also suggested that the network characteristics of the connections between lower order hypercomplex cells and higher order hypercomplex cells are similar to those between simple cells and complex cells. In their hierarchy, a 'higher' stage of cells in the hierarchy tends to respond selectively to a more complicated feature of the stimulus pattern, and, at the same time, has a larger receptive field and is more insensitive to the shift in position of the stimulus pattern.

Today, we know that this model does not reflect the true connectivity of the mammalian visual system, as there are several experimental data contradictory to the Hubel and Wiesel model (such as evidence for monosynaptic connections between LGB and complex cells). These findings, however, do not completely contradict the hierarchy model, if we consider that the hierarchy model represents only the main stream of information flow in the visual system. Moreover, even though the Hubel and Wiesel definitions of stages do not match later findings, there is much recent evidence to other, more complicated, hierarchical characteristics of the visual system.

In [15] from 1991, Felleman and Van-Essen describe in detail a summary of the defined cortical areas associated with vision (and with other modalities) in the macaque monkey, and a summary of their inter-connections. This work deals with 25 neocortical areas that are predominantly or exclusively visual in function, plus an additional 7 areas that are regarded as visual-association areas on the basis of their extensive visual inputs. Felleman and Van-Essen report of 305 connections among these 32 visual and visual-association areas , which is 31% of the possible number of pathways in such a complex system (they estimate that the actual degree of connectivity is higher and closer to 40%). With such a huge and complex system (that contains parallel and recurrent pathways) the simple and somewhat naïve model of Hubel and Wiesel may seem obsolete. However, although the feed forward model does not describe the true connectivity in the visual system, Felleman and Van-Essen conclude that the notion of hierarchical processing is still very relevant to the visual system. As long as each pathway well-defines levels of processing, it can be considered hierarchical. Figure 1 presents the network of cortical centers described by Felleman and Van-Essen.

Since in this work we deal with the specific visual task of object classification – a single processing pathway – there is an obvious biological reason in the usage of a hierarchical model (we ignore other visual functionalities that may interact with the different stages of the process and create loops in the network).

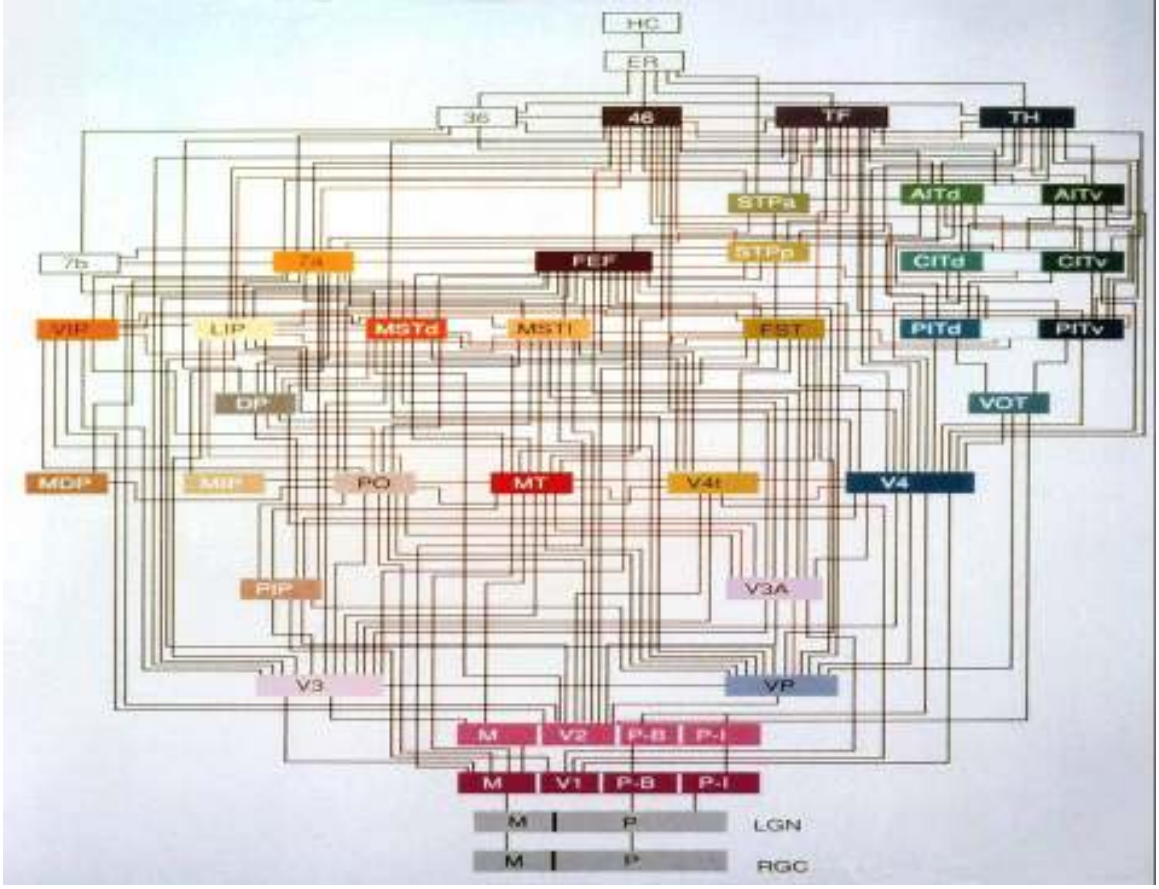

Figure 1: This hierarchy shows 32 visual cortical areas, 2 subcortical visual stages, plus several nonvisual areas. These areas are connected by 187 linkages, most of which have been demonstrated to be reciprocal pathways

3.1.2 V1 Characteristics

In addition to their research of the hierarchical nature of the cat's visual system, Hubel and Wiesel also studied the responsiveness of the cat's primary visual cortex (V1) to different features. In [16], they describe how simple and complex cells recorded in the cat's striate cortex respond strongly to oriented bars. They found that simple cells have small receptive fields with strong phase dependence, that is, with distinct excitatory and inhibitory subfields, whereas complex cells have larger receptive fields and no phase dependence. This led Hubel and Wiesel to propose a model (in [13]) in which simple cells with neighboring receptive fields feed into the same complex cell, thereby endowing that complex cell with a phase-invariant response.

Experiments showed ([17]) that young kittens that were grown in an environment deprived of visual stimuli of certain orientations (such as a room with only horizontal lines painted on it walls), grew up to have reduced sensitivity to stimuli of this missing orientation. The fact that the ability to see itself was hindered (and not only more complex, higher functions such as recognition), teaches us that the entire visual system may be dependant on V1 as Hubel and Wiesel described it, including possible object recognition pathways; the only input to the higher level areas is essentially via V1. Another important conclusion of these experiments is that V1, as well as higher order areas, depends on learning.

3.1.3 Evidence for Fragment Detection

Although the unique response of V1 to bars and edges may be considered as response to very simple and primitive image fragments, it does not mean that in general neurons in the visual system

are tuned to recognize image fragments. Experiments show, however, that in many cases cells respond preferentially to complex patterns.

Cells found in macaque Infero-Temporal (IT) cortex, the highest purely visual area in the ventral visual stream, and thought to have a key role in object recognition ([18]), are tuned to views of complex objects such as faces: they discharge strongly to a face but very little or not at all to other objects [19]. A hallmark of these cells is the robustness of their responses to stimulus transformations such as scale and position changes. Such cells, which may be considered as 'grandmother cells', somehow 'know' to recognize certain objects and differentiate them from others. There is quantitative psychological [20, 21, 22] and physiological [23, 24, 25] evidence that units tuned to full or partial views are probably created by a learning process, and that the view-invariant output may be explicitly represented by a small number of individual neurons [23, 25, 26].

In [27], Kobatake and Tanaka studied the responsiveness of neurons in the IT cortex of anesthetized macaque monkeys using single-cell recordings. They found out that these neurons' responsive fields are large and centered on the fovea. In addition, these neurons showed (as in the researches mentioned above) invariance to position within their responsive fields and moderate invariance to orientation and size. Kobatake and Tanaka further investigated the nature of the shapes that the neurons responded to, using a systematic image reduction method. For each neuron, they started by finding the natural image that stimulated the highest response. Once such an image was chosen, a series of reductions to it were made in order to find the simplest and most primitive features in it that would stimulate the same response. The final image that could not be broken or abstracted any further, encapsulated the 'essence' of the original natural picture that evoked the highest response. This abstract image was called then the neuron's critical feature.

What is most relevant to this work is the nature of the critical features found by Kobatake and Tanaka. The resulting critical features were images of different complexity. The 'visual complexity' (in terms of image specificity) of these critical features was what one could call Intermediate Complexity (IC). In many cells, the response was more sensitive to some local transformations than to others. Another interesting part of Kobatake and Tanaka's work was that they managed to organize the critical features in a columnar structure similar to that of V1 features.

This work tries to describe a model that is based on all these findings with what we know of the hierarchical nature of the visual system.

3.2 Computational Background

This section will provide some background on the computational aspects dealing with object classification using Artificial Neural Networks (ANNs). In addition, it will present the computational algorithm for constructing feature hierarchies (image fragments) for object classification.

3.2.1 Artificial Neural Networks in Classification

The remarkable computational capability of the human brain served as an inspiration to the development of the field of ANN. The potential in designing a computational model that would enable unlimited parallelism and scalability, combined with the nearly perfect and fast performance of the human brain brought scientists to investigate the principles of neural networks and to try and imitate them.

Already in 1943, McCulloch and Pitts ([28]) presented a very simple model of a computational unit that would function as an abstract neuron. This model was a discrete-time binary threshold unit, defined by a threshold θ and a finite set of weights $\left\{w_j\right\}_{j=1}^{n}$ (n is fixed for each neuron separately), that computed a threshold function ($sign$) over a weighted sum of its inputs at a current time-step t:

$$output\left(t+1\right)=sign\left(\sum_j w_j \cdot input_j\left(t\right)-\theta\right),$$

while throughout this work:

$$sign\left(x\right)\triangleq\begin{cases}1 & \text{if } x\geq 0\\ 0 & \text{otherwise}\end{cases}.$$

Networks of such McCulloch-Pitts neurons that were called perceptrons (feed-forward networks composed of layers of neurons between the inputs the output neuron), and the question - how to find appropriate weights for particular computational tasks, were subject to great scientific activity since the 60's.

In 1962, Rosenblatt ([29]) proved for the simplest class of perceptrons without any intermediate layers the convergence of a learning algorithm – a way to change the weights iteratively so that the desires computation was performed. However, in 1969, Minsky and Papert ([30]) proved that there are functions that cannot be computed without intermediate layers (such as XOR).

In 1974, Werbos ([31]) suggested a learning-algorithm for multi-layered perceptrons that was proven to converge – back-propagation (it was later re-discovered, in [32, 33], by Rumelhart et al.). Back-Propagation Networks (BPNs) are feed-forward perceptrons that implement a global gradient-descent algorithm, performed on all the parameters of the networks (all the weights). Back-Propagation is a supervised learning algorithm – a learning algorithm that takes as one of its inputs the network's requested, true, output (as a feedback on the network's function). With this information in hand, and given that each neuron is a McCulloch-Pitts neuron and that entire network implements a differentiable function, one can compute the derivative of the error function for each layer iteratively, and find the optimal correction to each of the network's weights. BPNs perform very efficiently for many learning problems. Particularly, since BPNs are designed for real inputs (for the gradient-descent), they serve as natural candidates for object-classification problems (since images are real inputs). However, no biologically plausible model implementing BPNs was ever proposed. Moreover, the gradient-descent converges to a local minimum which is often far from the optimal solution.

Another ANN algorithm for learning is a Kohonen-map ([34, 35]). Kohonen-maps implement an unsupervised learning algorithm – a learning algorithm that does not get any feedback on its performance or on the network's performance. In this algorithm, trained neurons correspond to clusters of example vectors (inputs). When used for classification, a Kohonen-map neuron's critical feature is obtained as the average of the vectors represented in its cluster. Since Kohonen-maps implement an unsupervised learning algorithm, the resulting features are not always relevant for the requested task, including object-classification tasks. Therefore, a classification function based on such features may not be efficient or at least include much irrelevant, wasteful, information.

Both BPNs and Kohonen-maps converge to features that are not training-images' fragments. Therefore, these methods ignore the knowledge on the superiority of IC features over other features. In addition, these methods search in the almost infinite space of possible features instead of narrowing down the possibilities to only fragments of the few given examples. For these reasons, in

addition to the reasons already mentioned, these two algorithms do not provide a possible framework for the purpose of this research.

It is important to note that since the McCulloch-Pitts neurons and the perceptrons were developed, a few additional ANN models emerged. Later models tried to supply abstractions which are more similar to real neural activity. Some models modulate the neuron's output – by controlling the firing rate, while others develop its computation – by integrating inputs over time and space ([36, 37] for example). These more complicated models serve indeed as better imitations of real neural networks. Yet, although these models are more accurate for describing an individual neuron, they are not convenient to use in a system level learning algorithm due to their high complexity.

3.2.2 Non-Biological Fragment Based Hierarchical Classification

Based on the conclusions from the Fragment Based Classification (FBC) approach ([6]), Epshtein and Ullman suggested in [7] a computational algorithm for constructing class-specific feature hierarchies (of image fragments. A feature hierarchy example is presented in figure 13, section 5.2) for object classification. This algorithm addresses two separated problems: how to select a hierarchical set of fragments that will be useful for classification, and how to construct a classification function that is based on these fragments.

The desired feature-hierarchy should be one that maximizes the information delivered on the class, while minimizing the redundancy. The FBC approach supplies an efficient way to choose a set of informative fragments. This approach is extended in this work in an iterative way to break each fragment to its most informative counterparts. The result of this procedure, ending in a hierarchical set of fragments, is an algorithm that achieves better results than the non-hierarchical algorithm which uses holistic fragments. The main reason for this improvement is that detecting a fragment by the detection of its counterparts enables reasonable flexibility in shape and size. The detection of an eye in a face, for example, can be improved if it will rely on the detection of a pupil, an eyelid and an eye corner; although the relative locations of these fragments and their inter-distances vary in different eyes, their existence within some boundaries should mostly point on the existence of an eye.

As in most classification schemes, this algorithm requires a training stage in which the classifier is constructed using data gathered on a set of class and non-class images. Only then it is possible to test the efficiency of the selected features for classification, using novel images – in a test stage. Since the hierarchy and its various parameters are determined only at the end of the training stage, this algorithm is an off-line algorithm. The entire training process is performed in four stages: initial fragment selection, fragment-hierarchy construction, classification parameters optimization and the construction of a classification function. Following is a brief description of these stages.

The selection of initial fragments is similar to the process presented in [6]. As a first step, a very large 'bank' of feature-candidates is created by the extraction of many fragments from the training set images – candidate fragments are 'cut' is many possible sizes and locations. Using a detection threshold θ_i for each candidate (with normalized cross-correlation as a similarity measure), a binary random variable $f_i(\theta_i)$ is defined for the detection or miss-detection of the fragment in each image. Another binary random variable, C, represents the class content or non-class content of each image. For each fragment (i), its mutual information (MI) with the class is measured over all images (I):

$$\mathrm{MI}\left(f_i(\theta_i);C\right)=\sum_{\substack{f_i(\theta_i)=\{0,1\}\\ C=\{0,1\}}} p_I\left(f_i(\theta_i),C\right)\cdot\log\left(\frac{p_I\left(f_i(\theta_i),C\right)}{p_I\left(f_i(\theta_i)\right)\cdot p_I(C)}\right).$$

After this number is maximized for each fragment by changing its threshold, a greedy algorithm is used iteratively to choose the initial fragments. In each iteration, a single new fragment is selected to increase the mutual information of the fragment set by maximizing the minimal addition in MI with respect to each of the already-chosen fragments. This procedure ends when the new information gained does not increase over some threshold.

To extract a new level of features in the hierarchy, a subset of the training set is chosen. For each existing fragment, a new training set is chosen to challenge its efficiency in classification. Therefore, the new training set contains all images in which the fragment is detected (class and non-class), with a small addition of images in which the fragment was almost detected. The purpose of creating such a new training set is to assist in finding sub-fragments that will successfully detect additional examples that were not captured by the original fragment. Once this training set is created for each fragment, a new level of sub-fragments can be extracted in a similar manner to the way the first level features were chosen. Sub-fragments were allowed to be taken from only in a limited area around the spot where the fragments were found (or almost found). As long as breaking a fragment increases the delivered information, this procedure continues recursively.

For each fragment and sub-fragment, the extraction process determines an allowed Region Of Interest (ROI) within which the fragment is searched. The size of each ROI is determined to optimize the classification. In a top-down manner, each ROI size is determined to maximize the MI of the relevant fragment and the class. A size too small, will cause sometimes the fragment to be missed due to its falling outside, and a size too big will cause more False-Alarms (FA). Once a hierarchy's node ROI was fixed, its sons' ROIs were chosen to optimize their affect.

Last, the classification function is described. First, each of the lower-most fragments in the hierarchy was searched in its ROI using cross-correlation measure. Then, in a bottom-up manner, the measure-of-detection of each super-fragment was determined using a weighted sum of its sons' measures-of-detection. The weights of each hierarchy node are calculated and optimized in the following way: First, with some fixed weights, the optimal positions of the node and its sons are found. Then, in these locations, the optimal weights are computed using back-propagation. With the new weights, again the positions are improved and so on until the weights converge to some local maximum.

This algorithm provides a strong motivation to the development of a semi-biological classification model that uses hierarchies of image fragments. Most of the calculations in this algorithm, however, cannot be translated in a way that will be biologically-plausible.

3.3 Some Previous Work (Artificial Neural Network Models)

This section reviews very briefly a few attempts done so far to use ANNs specifically for the task of object classification.

3.3.1 Non-Hierarchical FBC Model

In [12], Levi presents a biologically-plausible model that is inspired by the FBC approach, fitting the algorithm presented in [6] to the constraints of an ANN and a biological system. This

algorithm is an on-line algorithm; the entire network is complete and capable of performing classification tasks right from the beginning, and only its parameters and performance improve as the training proceeds. The ending of the training process is therefore somewhat arbitrary – at some point where no significant improvement is achieved.

The model presented in Levi's work consists of 2 layers of neurons: a fragment layer that is divided into columns and that covers the entire image area, and a decision layer which consists of a single neuron (for a single task). Each of the fragment neurons represents a single learned image fragment in its position, which is encapsulated in its weights in a procedure called neuronal imprinting. Neuronal imprinting is a way to determine the weights of a neuron in one shot, and by this to preserve a specific current set of inputs. This method is also used in the model described here, and will be explained separately.

The procedure for fragments selection is similar to the one described in [6]. In each training epoch, a new fragment is learned and replaces the 'worst' existing fragment. The fragment value function used is the fragment's Merit Value (MV) – a way to estimate a fragment's contribution to an existing set of fragments. The MV for a fragment f comparing with a set of fragments S is calculated using the Classification Value (CV) and the Redundancy Value (CV):

$$\mathrm{MV}(f,S)=\mathrm{CV}(f)-\mathrm{RV}(f,S).$$

The two value functions used are (C is a binary random variable for class existence):

$$\mathrm{CV}(f)=\mathrm{MI}(f;C)$$

$$\mathrm{RV}(f,S)=\max_{f'\in S}\mathrm{MI}(f';f)$$

These two value function are estimated through the usage of a set of lateral connections between neurons in the fragment layer in a biologically plausible manner.

The decision layer comprises of a single neuron that is connected to all fragment neurons, and implements a naïve-base classifier. The neuron's weights are learned by a local synaptic-adaptation rule.

After its training, the network achieved good classification results, competing even with those of the non-biological model described in [6]. Moreover, the features extracted by the two implementations were quite similar. Levi's work provides several ideas that are used in the model described here, including neuronal imprinting. It does not, however, explore the possibility of using a hierarchical set of features which is the goal of this work.

3.3.2 Hierarchical Models

The construction of a biologically plausible hierarchical model for classification seems important due to its relevance to cortical behavior. Over the years, several researchers have proposed ANN models for object classification that sometimes tried to imitate real neural activity.

In [8], Riesenhuber and Poggio constructed a model for classification and recognition. Their model uses a MAX-like operation applied to the inputs of neurons in the model, instead of the more common weighted summation. This operation may be beneficial for creating size invariance, and to overcome misclassification in case of multiple stimuli in the receptive field. However, in this work, Riesenhuber and Poggio created a model that does not extract class-specific features, but instead uses a fixed, pre-learned hierarchy of integral features. The only varying part in the model is the decision function of the last hierarchy in the network. This simple model, therefore, cannot be used

to classify a wide range of complex classes, and will fall short in performance compared to a class-adaptive model in challenging tasks.

LeCun et al. (in [10]) suggested a model for the task of classifying handwritten digits that does contain the extraction of class-specific features. However, this model uses back-propagation and weight sharing (groups of neurons sharing the same weights) – two techniques that are not biologically feasible. Moreover, the model's features (encapsulated in the synaptic weights) are random in the beginning and updated using back-propagation to improve performance by local optimization. The result is that the algorithm converges to a local minimum of the network error, which is related to a local maximum of the mutual information function, achieved by the converged features. In [38], Vidal-Naquet showed that informative fragments taken from images of a class, when used as features, contain in most cases much more mutual information. Furthermore, when using the same local optimization algorithm and starting with a set of image fragments (instead of random start points) there is not much improvement in the performance. This means that the mutual information of a set of image fragments tends to be close to a local maximum. Therefore, it is not appropriate for working with image fragments.

The work of Fukushima in [9] presents a nice implementation of a mechanism similar to what is seen in simple and complex cells: pairs of layers are connected together to provide a degree of invariance to position. Fukushima's work, however, raises problems similar to those described above related to the MAX model and to biological plausibility.

In [11], Rolls and Milward present the VisNet2 model, which investigates some aspects of invariant visual object recognition in the primate visual system. VisNet2 imitates key functionalities of primate object recognition in order to investigate cortical behaviors such as short-term memory and clusters of firing neurons (information delivered by the joint activation of neurons). It is much more biologically detailed than the model described in this work. Nevertheless, the purpose of this work is to explore specifically the potential of hierarchical FBC within the boundaries of some biological restrictions rather than to model real-life cortical function.

4 Hierarchical Artificial Neural Network Model for Fragment Based Classification

This chapter will describe in detail the model and algorithm proposed in this work. In fact, the chapter describes two model variations that were investigated, and which share most of the network's properties, each one with its own variants.

4.1 Initial Pseudo-Biological Restrictions on the Model

The model described in this work tries to obey several principles and restrictions in order to keep its biological plausibility, in an effort to suggest a mechanism of action of human neural object classification. Since the level of modeling here is the system level (rather than a cellular or a molecular level), the focus is on biologically plausible network behavior and not on the exact modeling of the neurons. The main restrictions we adopted in designing this model were the following:

- **Distributed, Local Computation** – In order to model neural activity, we had to avoid global calculations that get inputs on large parts of the network or that output changes of such large parts of it. In the brain, no calculation is done by some central 'processor', and all the calculations are distributed among many neurons. In the same way that a single neuron communicates only with a subset of the neurons in the network, and most extensively with its neighbors, we allowed each neuron in our model to use only local information from a set of neighboring neurons and to affect a limited number of target neurons (as there are axons in the brain that connect over extended distances). The requested computation, object classification in this case, is the result of many neurons acting simultaneously. In our case, the output is simply the result of one local computation done in a specific neuron or a small set of neurons.

- **On-line Learning** – During the training process of the network, we restricted ourselves to using only the memory capacity of the network itself. In the same way that our brain has no central processor, it also has no separate and infinite memory bank; all information contained in the brain is stored in neurons in the form of connections strength and a few other dynamic properties. In our model, therefore, we had to keep a 'short term memory' rule which means that we may not store information for future calculations exceeding our finite, fixed sized storage volume. Each neuron has a fixed set of parameters, and their values express the neuron's memory. The sum of all these parameters is the total memory space we could use in every step of the training process.

  Another aspect of on-line learning is the need to be able to stop the training at any given moment and to provide the complete network for queries. As no person ever 'shuts down' to process new information, our algorithm had to construct the complete network with all of its parameters from the beginning and to only update them gradually, step-by-step, as more and more training images are viewed, in a constant improvement. Obviously, if tested at the beginning of the training process, our network would perform quite badly. The end of the training process is reached simply when the network reaches its best performance and stops to improve.

4.2 Model Overview

The neural network model presented in this work is trained to perform object classification on a single class of objects at each training procedure. It uses two sets of training images for this process – class object containing images (training class images) and images that do not contain any class object (training non-class images). Using feature selection schemes that are based on the Fragment Based Classification (FBC) method, the networks achieves pretty good classification results. The efficiency of the trained network is tested using another two sets of test images that are distinct from the training sets – test class images (class object containing images) and test non-class images (images that do not contain any class object).

4.2.1 General Structure of the Network

The skeleton of the network is a standard perceptron which is an Artificial Neural Network (ANN) comprised of layers of feed-forward McCulloch-Pitts neurons ([28]) as described in section 3.2.1. This network contains three *learning layers* of neurons (plastic layers). The first layer, called L1, is the first *fragment layer*. Each neuron in L1 corresponds to a single training image fragment used as a classification features. L1 is organized in a columnar structure, such that all the neurons within a column cover the same retinal region and together they define a 'feature bank' for the corresponding area in the image. The second layer, L2 is another fragment layer. Each L2 neuron, connected to a patch of L1 neurons, represents a larger fragment than those of L1, and corresponds to a superposition of L1 features. L2 is also organized in columns similarly to L1. The third and last layer in the network is the *decision layer*. Containing only a single neuron, this layer generates the network's output. The decision neuron is connected to all L2 neurons. Its output is a number in the range [0, 1] that represents the network's classification of any given image (close to 0 means non-class and close to 1 means class).

This network gets its inputs from three additional image processing *static layers* (that do not change throughout the learning process). The first of these layers is the input layer which represents the image intensity values. The second of these layers, called V1, filters the input image through a set of simple line and edge filters and presents at each pixel the result of this filtering. In this way, the image is no longer represented by intensity values, but rather by these atomic features. The last of the three image processing layers is V1'. This layer sums neighboring V1 neurons of the same type. This local summation produces local position invariance and reduces the processed image size while keeping the filters representation. The output of V1' is input to the first fragment layer (L1).

Figure 2 shows a diagram of the network's skeleton. In the two models described in this work, this network's skeleton is maintained. The differences are in the feature selection procedure and the classification procedure.

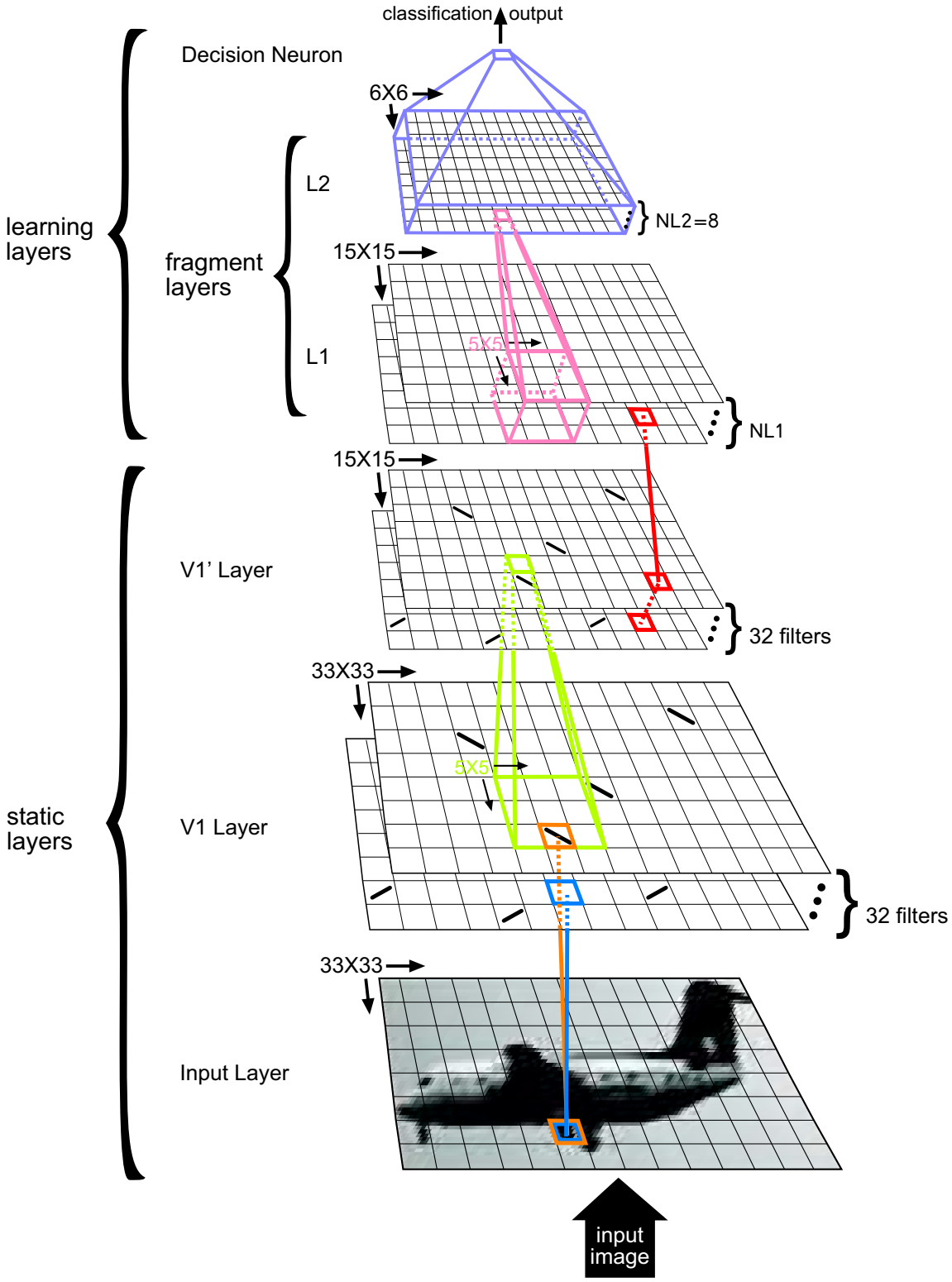

Figure 2: The network's skeleton: three static layers that process the input image, followed by three learning layers that implement the feature selection and decision making algorithm. The length of a L1 column, NL1, varies in the different variants of the algorithm

4.2.2 General operation of the model

In general, the model proceeds by alternating between two main stages: a feature selection stage and a calibration stage. Initially, the model uses class examples to construct complete feature hierarchies. Small image regions from the class examples are chosen as features of the L1 layer, and

activated L1 units are used to select fragments that construct L2 features. The network then goes through a calibration stage. During this stage the network does not change its connectivity, but it uses additional class and non-class examples to determine the relative contribution of units at the different levels to the classification results and to update and improve the network's dynamic parameters. Units with the smallest contribution to successful classification become modifiable, and during the next feature selection stage some of them are replaced by new features, chosen from new class examples. Units from both the L1 and L2 layers may be replaced during a feature selection stage.

4.3 Model Description

Following is a description the two model variations we investigated. First, the three static layers are presented, then some features and characteristics are described and last we detail the structures and function of the two learning models separately, each with its own variants. Throughout this section, wherever there is a biological justification to some specific aspect or feature of the model, it is presented in detail.

4.3.1 Static layers – the Input layer, V1 and V1'

During the training and test stages of the network's action, class and non-class images are alternately presented to it for learning and for classification. However, the data that the learning network receives is not the images themselves but a processed version of them. The first three layers of the model implement a static ANN with fixed weights and therefore, in relevance to classification, with fixed features.

This initial stage in the network is inspired by the function of first processing stages in the human visual system. Using a set of lines and edges in a range of orientations, the presented image is transformed into data similar to the information expressed in the primal visual cortex.

4.3.1.1 The Input Layer

The input layer presents the observed image to the network after some minor adaptations, and is not really one of the ANN layers. Every image presented to the network is arranged in a matrix that contains intensity values. The size of this matrix is fixed, and is the entire network's visual field. The input layer performs a small manipulation on the image. Taking some grayscale image with some positive intensity values ($I_{m\times n}$), it transforms it to a matrix of real values with an average of 0 ($I'_{m\times n}$):

$$I' = I - \frac{\sum_{i=1\ldots m, j=1\ldots n} I_{i,j}}{m \cdot n}$$

and then to a matrix with a variance of 1 ($I''_{m\times n}$):

$$I'' = I' \cdot \frac{m \cdot n}{\sum_{i=1\ldots m, j=1\ldots n} \left(I'_{i,j}\right)^2}.$$

The purpose of this procedure is to reduce the effect of luminance changes on the classification task. This computation is general and not class specific. When the luminance and color effects are decreased, the shape and the pattern of the image become the most significant features. Since we

would want this model to perform well on classes that have varying colors (such as cows that can be white, brown or spotted) and to perform well on images taken with varying light conditions, this procedure is essential. From a biological point of view, this stage that normalizes the entire image may simulate aspects of light adaptation in the eye and the retina (as described in [39], for example).

Throughout this work we chose to work with square images of size $m \times n = 33 \times 33$ (pixels).

4.3.1.2 V1 Layer

The output of the input layer, a real valued matrix, is the input of the second layer, called V1. V1 layer has the same size the input layer (33×33), covering each pixel of the input image separately, together with a third dimension that turns V1 into a 3-dimensional matrix. Each pixel in the input layer corresponds therefore to a vector along the 3rd dimension of V1. This vector is called the *V1 column* that corresponds to that pixel, and we therefore say that V1 is organized in a columnar structure.

Each V1 column delivers information on the similarity of the corresponding pixel's environment and a set of fixed V1 filters. These filters, which may be seen as generic, atomic features or fragments, are lines and edges in several orientations. The set of filters used in this model contains 32 filters: 16 line filters of size 7×7 (8 filters of dark lines on a bright background in angle increments of 22.5º and 8 similar filters of bright lines on dark background) and 16 edge filters of size 6×6 (in angle increments of 22.5º). The filters used were drawn using the convolution of binary lines and edges with a Gaussian ($\sigma = 0.85$). A few of the filters are shown in figure 3.

Figure 3: A few V1 atomic features (or filters) – edges and dark\bright lines

The result of this 3-D structure of V1 layer is that for each image presented to the network, V1 calculates its similarity at each location to all the 32 filters. The length of each V1 column is, therefore, 32 (and V1's size is $33\times33\times32$) and each neuron in a V1 column measures the similarity between its corresponding filter and the relevant patch in the image (as it is represented by the input layer, and centered around the pixel corresponding to the entire V1 column). The similarity measure we chose to work with is the normalized cross-correlation between the filter ($F_{n\times n}$) and the relevant patch ($P_{n\times n}$) in the input layer's image:

$$\frac{\frac{1}{|P|}\cdot\sum_{i=1\ldots n,\,j=1\ldots n}\left(P_{i,j}-\overline{P}\right)\cdot\left(F_{i,j}-\overline{F}\right)}{\sigma_P\cdot\sigma_F}$$

while $|P|$ is the patch's total size -

$$|F|=n^2,$$

$\overline{P}$ and $\overline{F}$ are the patch and filter averages correspondingly -

$$\overline{P}=\frac{\sum_{i=1\ldots n,\,j=1\ldots n}P_{i,j}}{|P|}$$

and σ_P and σ_F are the patch and filter standard-errors correspondingly -

$$\sigma_P=\sqrt{\frac{1}{|P|}\cdot\sum_{i=1\ldots n,\,j=1\ldots n}{P_{i,j}}^2-\overline{P}^2}\ .$$

The result of this similarity measure is always in the range [-1, 1] such that positive values indicate measure of similarity and negative values indicate reverse similarity.

A characteristic of V1 layer with these features and with this similarity measure is that in any column, for every presented image, there will always be pairs of inverse numbers (positive and negative) with the same absolute value. The reason is that for every filter there is a matching reversed filter with bright intensities instead of dark ones and the other way around. Since later on in the model development we sometimes want to avoid negative values as inputs, this result is beneficial to us; taking only the positive results for each patch and each filter does not reduce the information conveyed on the image.

The second operation of V1 is the selection of areas of interest in the image and the elimination of minor responses to the filters. Using a Canny Edge Detector (CED, described in [40]), our model searches for areas that deliver a large amount of information on lines and edges. The similarity measures of all V1 neurons and all V1 filters are then filtered through this mask (binary separation to informative and non-informative areas) so that only similarity measures in informative areas remain (the rest of the measurement are zeroed). As for the remaining measures, they are processed by a filter that allows only significant similarities with an absolute value of over some *V1 threshold* (we worked with V1 threshold of 0.2) to survive.

From a biological point of view, the V1 layer model simulates some characteristics of cortical V1 units. More specifically, it resembles the activity of simple cells in the primal visual cortex as described in section 3.1.2. The columnar structure, the small receptive fields, the lines and edges detectors, the Gaussian filters and the luminance invariance are all believed to be actual characteristics of neurons in this part of the visual pathway [13, 14]. The CED, a computer algorithm, simulates a form of primitive image segmentation or contour detection. It is used to differentiate between lines and edges of the image's main object and of the background (which are

identified as not-connected to the main object and are therefore ignored). Similar contour detections were found in V1 ([41]).

4.3.1.3 V1' Layer

The output of the V1 layer, a real valued matrix, is the input of the third layer, called V1' (V1 prime). The structure of V1' layer is very similar to that of V1 layer. With a 3-dimensilnal structure and a columnar organization, it maintains the information organization of responses to the 32 filters used in V1. However, while every *V1' column* (a V1' vector along the 3rd dimension) is of length 32 and corresponds to each of the V1 filters, the area size of V1' (its 1st and 2nd dimensions) is smaller than that of V1. This is the first step of a series of steps of reducing the *layer area* (the spatial dimensions of each layer of the network, which correspond to different receptive fields).

There are two reasons for these area reductions. The first reason is a computational one – since our model implements a hierarchical system that ends in a single decision neuron, we try to gradually decrease the number of neurons in each layer. The second reason rises from our effort to simulate the visual system – a fundamental feature in the visual pathway is the increase, as one examines higher visual areas, in the receptive fields of neurons. Our gradual reduction of the layer areas will go together with the gradual increase in our neurons' receptive fields.

The layer area we chose for V1' is 15×15. Each V1' neuron sums the responses of 5×5 V1 neurons that correspond to the same V1 filter. The result of this size reduction procedure is a $15\times15\times32$ sized layer. It is important to notice that the separation to all V1 filters is maintained, and that each V1' neuron only sums an environment of V1 neurons of the same 3rd dimension position (in V1). The resulting receptive field size for V1' neurons is 5×5 pixels which are about 2.3% of the entire visual field.

Besides the layer area reduction and the receptive field increase, this summation of V1 neurons achieves another important goal. Due to the summing operation, V1' allows some measure of tolerance to translation, or position invariance which is an important characteristic in a classification scheme. In addition, it is also desired in our model since it is a feature of the visual system. To complete the work of V1' to achieve position invariance, we enforced the filtering through areas of interest in V1 and through the filter of V1 threshold. Our assumption was that since only 'significant' similarity measurements survive (i.e. high measurements from informative areas found by the CED), a feature that appears in different areas in the patch will evoke similar responses in a V1' neuron while a collection of minor responses that may sum to the same value will be ignored (and thus will not be able to disguise as a single significant response). Preliminary tests we performed of measuring V1' responses to an image moved around showed that a certain degree of position tolerance is obtained, as neurons' responses remained in many cases at similar levels as long as the image moved within a short distance. Moreover, the images we used to test our entire network were not perfectly cut to reveal the same proportions of the class objects (and so to minimize the position variance) and as will be shown later, the classification results were still good. Another important result of our selection of V1' receptive field size is that neighboring V1' neurons have receptive fields that overlap – this characteristic of our network contributes to its position invariance, since it enables the network to fit the best receptive field to each feature; as long as a certain feature tends to move around the center of the receptive field, it will be identified by the network.

The biological justification behind the work of V1' is its contribution to position invariance in a way that resembles the work of complex cells in the primal visual cortex and their contribution to phase invariance as described in section 3.1.2 ([13, 16]).

The output of V1' is the input to the next layer of the model, L1, and to the learning layers of the network.

4.3.2 Features and Characteristics of the Model

This section presents several properties and features that are shared by the two model variations we developed. After reviewing these properties and providing biological reasoning to some of them, we will be able to describe each of the variants separately (in the following sections).

4.3.2.1 Columnar Organization of the Fragment Layers and Their Structure

The first two layers of the plastic, learning part of the network are the fragment layers. Each neuron in these layers corresponds to a single specific feature located in a fixed part of the visual field. The method used in order to recognize a specific fragment with a specific neuron is called neuronal imprinting and will be described later. The features we work with are always training class fragments, which are picked on-line by a feature selection scheme.

The first fragment layer, called L1, gets its inputs from the last layer of the static part of the network: the V1' layer. The layer area of L1 is identical to that of the V1' layer (15×15 in our tests), and the two layers share the same receptive field sizes. Similarly to V1' layer, L1 is also organized in a 3-dimensional columnar organization. Each vector of neurons along L1's 3rd dimension is called a *L1 column*. All the neurons of the same L1 column are connected to the same neurons in the V1' layer; each such L1 neuron has synapses connecting it to the entire V1' column which is located directly 'below' it. That is, a L1 neuron in position (i, j, k) has exactly 32 synapses – one from each V1' neuron of the V1' column in position (i, j). As a result of this structure, each L1 neuron covers a receptive field of 5×5 pixels. Moreover, each such neuron receives the outputs of V1' neurons that correspond to all of the 32 atomic fragments. Therefore, a L1 neuron gets all of information our atomic filters found on an image fragment in the relevant receptive field. During learning, a subset of the 32 synapses from a V1' column to a given L1 neuron becomes effective (with a specific set of weights), and the L1 unit then becomes selective to a particular configuration of V1 features within its receptive field.

The columnar organization of the L1 layer separates L1 into 225 groups of neurons (each group is in a different L1 column – 15×15 columns) that get the same information on each image presented to the network. Since each L1 neuron learns how to identify a specific image fragment, we get 225 sets of image fragments. Each L1 column provides us then with a 'feature bank' of image fragments of its corresponding receptive field. Our feature selection scheme tries to exploit this redundancy of features to improve object classification. The L1 column length we chose is called *NL1*, and is not the same in both of the model variants we investigated.

The second fragment layer, called L2, gets all of its inputs from L1. Each L2 neuron is connected to all the L1 neurons contained in a *L2 patch*. In order to increase the receptive field size of L2 neurons compared with that of L1 neurons, each L2 neuron has synapses connecting it to all L1 neurons that are found in a small environment 'below' it. Using L2 patch size of 5×5 in our experiments, the result is that a L2 neuron has exactly $5^2\cdot\mathrm{NL1}=25\cdot\mathrm{NL1}$ synapses connecting it to 25 L1 columns. The practice of L2 patches goes together with a reduction of L2's area size. In the second step of reducing the number of neurons that cover the visual area and increasing receptive fields, we chose L2 area size of 6×6. This, as in V1' layer, results in overlapping receptive fields of L2 columns. Each L2 neuron in this case has a receptive field of 13×13 pixels which are about

15.5% of the entire visual field and, therefore, match with our interest in features of Intermediate Complexity (IC) (unlike L1 neurons that cover about 2.3%).

Similarly to L1, L2 also has a columnar organization that creates 'feature banks'. In a 3-dimensional structure, each *L2 column* (a vector of L2 neurons along the 3rd dimension) contains neurons that are all connected to the same neurons in L1. The length of L2 columns we chose to work with is called *NL2*, and we fixed it to be 8 neurons. This choice of connections between L2 and L1, that seemed natural when used between L1 and V1' layer, raises some problems. A L1 neuron that is connected to all the neurons in a V1' column gets totally different information from each one of them. Since each neuron in a V1' column is associated with a different atomic fragment, and since we want each L1 neuron to be able to 'remember' a complete image fragment, there is rational in connecting L1 to V1' in this way (a connection partial to the one we used will result in an incomplete description of the receptive fields to L1 neurons). This way, a L1 neuron performs some superposition on its inputs to construct an image fragment. In L2, the situation is not the same since L1 features are already image fragments and not atomic features, and so the information they convey may overlap. Since we want L2 neurons to correspond to image fragments, connecting them to L1 neurons that react to a few different image fragments over the same position seems redundant. The connectivity rule we used was therefore the following: although each L2 neuron has connections to 25 complete L1 columns, in practice it will have at any given moment only 25 active connections (one from each L1 column) while the rest of the connections will be inactive with a neuronal weight 0. The choice of the active connections, together with the entire management of the 'feature bank' of L2 is the work of the feature selection algorithms we implement in our model variants.

The columnar organization of L1 and L2, matches with neurobiological findings of Tanaka ([42, 43]) that point on columnar organization of neurons in some higher visual centers in the monkey brain.

4.3.2.2 Normalization of Neurons' Inputs

In the proposed model, whenever an image is presented to the model and before all of the networks' neurons compute their outputs from their inputs, we normalize each neuron's input vector – only for the learning layers (L1, L2 and the decision layer). That is, for a neuron in L1, L2 or for the decision neuron connected to j input neurons with corresponding j outputs $I = \left(i_1, i_2, ...i_j\right)$, we calculate the normalized vector $\hat{I} = \left(\hat{i}_1, \hat{i}_2, ...\hat{i}_j\right)$:

$$\hat{i}_k = \frac{i_k}{\|I\|_2} = \frac{i_k}{\sqrt{\sum_{t=1...j} i_t^2}}$$

as the input vector (for the scalar product with the neuron's weights).

The computational benefit of this procedure will become clear when we explain how we determine neuronal weights and calculate outputs. As for the biological support for this procedure, there is evidence to invariance to luminance in different parts of the visual system (such as [44]). This normalization is in addition to the global normalization we perform in the input layer. Although we perform this procedure at a single step for each neuron in contrast to biological findings, our interest is in the result and not in the details of the normalization procedure.

4.3.2.3 Neuronal Imprinting: One-Shot Learning

Neuronal imprinting, as presented by Levi in [12], is the biological model we use for implementing the FBC approach of using image fragments from the training set as features for object classification. Our motivation for using neuronal imprinting is to be able to capture in one-shot, at a single moment, a fragment of the image currently presented to the network. In neuronal imprinting, the entire set of weights of a neuron is replaced with a completely new one. Our need to use such one-shot learning rises from our limited memory restriction and from our interest in using actual image fragments.

Neuronal imprinting occurs whenever a specific neuron is chosen (in the feature selection process) to capture and record its current inputs instead of the feature it is currently associated with (which is discarded). The imprinting procedure is this: the chosen neuron simply replaces each of its neuronal weights with its synapse's current input. Without retaining the existing neuronal weight, each synaptic weight is set to this new value. The 'forgetting' nature of this procedure forces the feature selection scheme to deal not only with the question which features to chose, but also with the decision which features to abandon. The procedure of replacing a neuron's set of weights that corresponds to an image fragment with a new set is called re-imprinting.

The benefit of neuronal imprinting becomes clear when considering the calculation done in each neuron together with the application of input initialization (as discussed previously). Since neurons imprint normalized vectors and get as inputs other normalized vectors, and since the calculation a neuron performs is scalar product, we get a similarity measure between the inputs and the imprinted weights. The neuronal calculation becomes a product of two unit vectors. The imprinted feature will stimulate a maximal response, and other inputs will stimulate lower (and even negative) responses. The responses range will be [-1, 1] if negative inputs are allowed or [0, 1] if only positive ones are allowed. If the inputs of a neuron are image fragments (or inputs that represent image fragments such as in our L1 case), then the result of the imprinting procedure will be a snapshot of the relevant fragment. In the case of L2 in this model, the inputs are responses of L1 neurons to their fragments and may also be considered as a superposition of image fragments that can be imprinted.

In a systematic research of the visual recognition capabilities of pigeons, it was shown that after a single presentation of hundreds of images they were capable of distinguishing between old and newly presented images quite remarkably ([45]). Everyday experience also teaches us that in many cases a single exposure to an object, such as a person or a specific room, is enough to enable us to recognize and distinguish it. The term 'imprinting' is taken from the field of animal behavior, where the pattern of hatching ducklings that attach forever to the first object they see is well known ([46]). These examples provide hints for the ability of learning in the visual system and object classification pathways, in one-time. The existence of neurobiological mechanisms that enable very quick learning is supported by the findings in [47], where Markram et al. showed that in a very short time where only as few as 50 Action Potentials (APs) take place, significant synaptic changes were induced. All of these findings support the plausibility of a one-shot learning method such as neuronal imprinting.

4.3.2.4 Local Gradient-Descent for Neuronal Weight Update

Although the neuronal imprinting mechanism together with the input normalization mechanism forms a similarity measure for presented images and memorized fragments, the outcome of these measurements do not always provide good classification results. When a neuron is imprinted, only

some images that resemble (to some extent) its critical feature may cause a positive response. The imprinted fragment may be too general or too specific, or that it may be too similar to some set of non-class images, causing many classification errors. A desired change in a neuron's response can be achieved via controlled changes in its dynamic variables (i.e. the synaptic weights and the output threshold).

The problem we would want to address here is the problem of on-going neuronal update. Since each neuron in our model learns to recognize a certain fragment, one could think of each neuron as a classifier of that certain fragment. In order to improve a neuron's performance in separating images that contain its relevant fragment from images that do not, it is useful find which of the neuron's inputs deliver more accurate data and contribute more to the classification procedure, and to increase their weights. The solution we chose to cope with the problem of neuronal update is to allow synaptic plasticity according to a *weight update rule*.

Since we treat each neuron as its own fragment's classifier, the error function we would want to minimize locally is the distance between the neuron's scalar product of weights and inputs ($\bar{w}\cdot\bar{I}$) and the binary variable that measures the existence and non-existence of the relevant fragment in the currently presented image (f). Unfortunately, we do not have this binary variable and there is also no definition for the existence or non-existence of the fragment in an image. Since all of the neurons should perform positive detection, i.e. to respond positively for class images and negatively for non-class images, and since we imprint only training class images' fragments, a good estimation for f is the class variable that says if the currently presented image belongs to the class or not (C). The simplified error function we chose to minimize for a neuron is, therefore:

$$e(\bar{w}) = \left| C - \bar{w}\cdot\bar{I} \right| .$$

More complex measures of the neuron's performance are also worth considering (such as measures that also take into account the activity of the neuron's parent in the hierarchy). In the present work we focused on this simple measure. To achieve this minimization, we perform a gradient-descent along the derivative of $e(\bar{w})^2$ (we take the squared error rather than the absolute value), and in each update step we reduce a small percentage (α) of this derivative from the weights vector $\bar{w}$:

$$\Delta(\bar{w}) = -\alpha\cdot\frac{\partial e(\bar{w})^2}{\partial\bar{w}} = \alpha\cdot\left(C - \bar{w}\cdot\bar{I}\right)\cdot\bar{I} .$$

The parameter α is called the *weight learning rate*. We used different α's for different variants of our model.

The enforcement of an update rule may seem to interfere with the usage of neuronal imprinting and the FBC approach. However, in [38] Vidal-Naquet showed that when utilizing gradient-descent on initial image fragments, classification results tend to be significantly better then when starting off from arbitrary values. Moreover, the initial values established through neuronal imprinting limit our search space of features to only image fragments which is a much smaller space to look in than the global space of all possible features. To summarize, the neuronal imprinting procedure still contributes to the classification procedure, and the weight update rule only improves it. In addition to that, in order to use as our features' construction blocks unchanged image fragments, we made a rule that L1 features will not be affected by the learning rule and that L1 weights will be determined only through neuronal imprinting (since L1 features are directly linked to specific image fragments, we decided not to touch them). Other features, of L2 and of the decision neuron (which will be described later) represent superpositions of L1 fragments and may therefore be altered. The

consequence of this is that the relative importance of L1 fragments that contribute to a L2 neuron or to the decision neuron is changed; this does not really contradict the FBC approach.

The synaptic plasticity model we chose to work with is based only on local information at the neuron level, and may be seen as some form of Hebbian learning ([48, 49]); since the result of the weight update scheme we chose is that neurons that make less errors get stronger synapses, and since all neurons aim to fire together when a class image is presented, we end up with synapses that strengthen for neurons that fire in a synchronized manner on the presentation of class images. There are a few experimental results that support similar mechanisms of neuronal plasticity ([50, 51]).

4.3.2.5 Errors Measurement

In order to manage the feature selection procedure that connects neurons to their parents and to determine the priority of neurons that should be re-imprinted, there should be some way to measure a neuron's value. As discussed earlier (in section 4.3.2.4), the random variable that tells whether or not the currently presented image belongs to the class, is a good predicator for the neuron's desired output. It is therefore possible to evaluate a neuronal value measure using the error rate of the neuron in classifying images to class and non-class ones (through a neuron's standard output). We use for this purpose two counters for each neuron: one for the number of its misses and one for the number of FAs. Dividing each of these numbers by the total number of observed images gives the neuron's error statistics.

Counting the error rates of neurons is a simple local measurement that relies only on the performance of a neuron and its inputs. As for the class random variable, it can arise form a feedback signal fed to the network after each training image, and which reaches at the same time all the neurons in the learning part of the network. Therefore, the error measurement complies with our restrictions regarding biological plausibility.

4.3.2.6 Using Limited Memory - Replacing Less-Wanted Fragments

As stated before, one of the problems our model should address is the choice of neurons to undergo re-imprinting. This choice results in the loss of a feature or a set of features that were potentially used by the network, and therefore this is a risky selection that must be done cautiously (a set of features may be lost when a discarded fragment has impact on his ancestors). Due to the fact that both fragment layers L1 and L2 are organized in a columnar manner that creates 'feature banks' over the same receptive fields, we have a limited ability to discard features without damaging the network's performance too much. Moreover, we can always try to pick for re-imprinting neurons that have no impact on the decision neuron (due to inactive connections), although this information is not necessarily available to us. Using a relative value assessment of neurons, such as the error measurement presented in section 4.3.2.5 or any other measurement, we can rank the neurons in a layer or a column and chose candidates for replacement in an effective way. The feature selection schemes we present in this work try to combine value measurement together with assessment of neurons impact on the decision neuron to achieve a good selection method.

4.3.2.7 L2 Connections to the Decision Layer

The decision neuron in the final layer is connected to all L2 neurons (in our experiments this means $6^2 \cdot NL2 = 288$ connections). Due to the same reasons that we did not want L2 neurons to be affected by more than one L1 neuron of the same L1 column, we do not want two L2 neurons of the same L2 column to have impact on the decision neuron. The simple solution to this problem is to

implement the same rule we determined in L1: At most, a single decision neuron synapse of all synapses that connect it to a specific L2 column will have a synaptic weight different than 0. This is indeed the solution we used at the beginning for our first model variant, but in the second model variant we developed a more complex solution that is explained separately.

4.3.2.8 Training Epochs of the Learning Layers

In all the network's training model variants presented in this work, the entire learning stage is divided into iterations that are called training epochs. Each training epoch is comprised of two stages: performance evaluation and weight update stage, and re-imprinting stage.

During the first part of a training epoch, the performance evaluation and weight update stage, random class and non-class training images are presented to the network for classification and the neurons' performance (including the performance of the decision neuron which provides the network's output) is measured. The performance measurement may be, for example, error rates measure, but it may also be other measures of the on-going activity of the network. At the same time, the weight update rule is invoked in all the relevant neurons. This stage ends after a fixed number of images were presented to the network (an equal number of training class and non-class images). The chosen number of images, called the *weight update loop length*, should be long enough to enable neuron weights to stabilize and to enable the gathering of enough data on the network's performance. The weight update loop lengths we chose differ in the different model variants.

The second stage of the training epoch, the re-imprinting stage, is the part of the algorithm where our feature selection scheme takes place. In this stage, all of the information that was gathered in the last performance evaluation and weight update stage and in some of the previous stages is processed together to decide which features should be kept and which should be discarded. The neurons that were chosen for re-imprinting are then re-imprinted with new image fragments. Last, all counters that should not be influenced by the past are initialized and the next training epoch starts.

Since our model should implement an on-line learning algorithm, there is no clear end to the training stage and to the training epochs. At some point the network should stabilize and reach a satisfying level of performance. Therefore, we designed our algorithms in a way that converges to a state where even if training epochs continue the classification performance will not be affected.

4.3.3 Model Variant I: Searching for the Best Features

The first model variant we constructed to perform object classification using the hierarchical network was a straightforward one, using error measurement to determine feature selection as well as for re-imprinting. We constructed a top-down feature selection algorithm that connects in each training epoch each neuron to its best children, and replaces the worst neurons in each column.

4.3.3.1 Variation I: Weights Update Rule Used in L2 and in the Decision Neuron

The threshold we chose for the L2 neurons and for the decision neuron was 0.5 – at the middle of their binary output range. For L1 we decided not to use a threshold in order to for it to convey maximum information on the level of detection of its imprinted fragments. Since we imprint only training class images that should cause neurons to fire for class images and not fire for non-class images, we wanted to avoid imprinting L2 neurons with negative values. We chose to ignore, therefore, negative outputs of L1 and to set such outputs to 0.

In the initialization stage, the network's neurons were imprinted using random training class images, and using different images for each layer. Since in all of the variants of this first model we fixed NL1 to be 8 neurons, each of the fragment layers required 8 images for its initialization, and a single image was used to initialize the decision neuron. The active sons for the decision neuron and for L2 neurons were chosen arbitrarily.

The performance evaluation and weight update stage of each epoch was separated into halves (that consisted of 400 training images together). During the first half errors were measured for L1 neurons only, while the decision neuron and L2 neurons had their weights updated using the gradient-descent scheme (the learning rate we used was $\alpha = 0.01$). In the second half, all neuron weights were fixed and errors were measured for all the neurons in the learning layers. The reason for this separation is that we did not want to measure errors of neurons before their weights converged. We observed that in many cases the update rule has strong impact on neurons' performance, and that measuring error rates at the beginning of this process may be misleading. L1 errors, therefore, were measured right from the start (where the update rule was not activated). Since L2 and the decision neuron's weights never ceased to change, we initialized their error counters at the beginning of every training epoch not considering whether they were affected by the last re-imprinting stage or not.

During the re-imprinting stage of each epoch, a hierarchical, top-down, randomized feature selection scheme took place. First, a way of ranking all the neurons in each L1 and L2 column (which will be described soon) was invoked and each neuron got an *error value*. Then, neurons that became candidates for re-imprinting were marked. The decision neuron was always marked, and so were the worst neuron in each L1 and the 4 worst neurons in each L2 column (at start we marked only one, but then in order to speed up convergence we changed to replacing four). At the same time, the best neurons in each L1 and L2 column were also marked. Going from the decision neuron downwards, each neuron that is marked for re-imprinting is saved from re-imprinting in a probability that equals its error value, which decreases as the neuron improves. Once a neuron was removed from the set of neurons that were candidates for re-imprinting, all of its sons were kept as well. However, since L1 neurons may have multiple parents, the only consequence of this definition is that as long as the decision neuron is kept, no other neuron is re-imprinted. The randomized decision was meant to achieve reduction in the network's plasticity as the algorithm converges. At the last stage, in a bottom-up manner, the marked neurons were re-imprinted. Neurons that were marked in L1 for re-imprinting had their weights replaced using 4 random training class images (no two neurons of the same L1 column were re-imprinted using the same image). Then, marked L2 neurons were re-imprinted using another training class image and were connected to their best sons (as marked previously). Last, the decision neuron was re-connected to its best marked sons and re-imprinted using a fifth image. After the network was re-imprinted, error counters of all replaced L1 neurons were initialized and the next training epoch began (all the other error counters were initialized anyway as mentioned before).

The outcome of this feature selection scheme is that each neuron acts in a greedy way to chose for itself its best possible child. The 'feature banks' of L1 and L2 are used merely as sets of candidates that, hopefully, improve all the time and are not necessarily fully exploited by the network at any given moment. At first, the network took a very long time to converge and its performance oscillated with time. The reason for this behavior was that each time the decision neuron or L2 neurons were re-imprinted, the entire process of tuning their weights took a long time and caused drastic and unwanted changes in the network. The solution we chose for this was to practically discard the re-imprinting procedure for these neurons and instead to leave their weights unmodified when re-connected to their newly chosen children. The rational behind this procedure is

that a 'bad' son which is replaced must have a low weight that can only improve. Since it is unlikely that an image class fragment will have a high FA rate and a low miss rate (as if it predicts the opposite of the class), 'bad' neurons must be such that have a weight close to zero (only in the unlikely case a neuron's weight may converge to a negative value). This was another reason for avoiding negative outputs in our network – allowing this may have caused more negative weights, and newly re-connected neurons would have started off a very low point in such a case. Indeed, throughout our experiments in this model we observed that converged weights were almost always positive. Convergence was achieved in this scheme due to the fact that after some time, after all the network's 'best' neurons were chosen and after a good set of L1 fragments was imprinted, changes occur only in insignificant neurons that have no impact on the network's output.

In order to rank neurons in each column and to calculate their error values for the randomized decision, we had to combine together the two error counters (misses and FAs) of each neuron. We compared several ways to calculate error values:

1. Average of the FAs counter and the misses counter
2. Weighted average of the FAs counter and the misses counter, using the relative weights of the decision neuron's two types of errors. The motivation for this approach was to give heavier weight to the network's weak side
3. Addition of the average between the two error counters and the distance between them. The motivation for this approach was to combine the neuron's error the amount of information it delivers. A neuron with one error counter high and the other low may have similar error average as a neuron that has both errors averaged. However, the first neuron in this case may be significantly less informative than the second one since it may be a neuron that just fires all the time or that never fires. A neuron with two error counters of similar value must 'know' something about the examined class
4. Addition of the weighted average between the two error counters and the distance between them. This approach is the combination of the second and the third ones
5. A single error counter's value. The chosen error is the one where the decision neuron scores lower. This approach is an extreme version of the second approach

The classification results did not show a significant benefit to any of these ranking methods. The error ranks we worked with were, therefore, the simplest one of the first option.

The classification results achieved by this scheme were satisfactory – worst than the results achieved by the non-biological algorithm in [7], but close enough considering the heavy restrictions we enforced on our model. However, as will be explained later in the results chapter, they did not teach us a lot on the efficiency of the feature selection scheme we designed.

4.3.3.2 Variations II and III: Testing the Necessity of a Weights Update Rule

Using the same scheme presented in section 4.3.3.1, we tried to test the network's performance when avoiding the weight update rule to see how efficient our pure feature selection scheme is using simple neuronal imprinting.

In one variation, we stopped updating the L2 weights using the gradient-descent weight update rule. Due to the fact that L2 neurons in this scheme do not need time for their weights to converge, we used the entire performance evaluation and weight update stage of each epoch for measuring L2 errors. The weight update loop length was decreased to 300 images. In this variation, L2 neurons imprint and re-imprint the outputs of their selected children and therefore correspond to a specific

combination of their children's fragments. Since our goal was to allow L2 to capture its inputs in each re-imprinting, we allowed L1 negative outputs to remain. In addition, we decided to remove the threshold from the L2 outputs and to let the scalar products in L2 deliver the similarity measurements between their inputs and their imprinted weights continuously.

In another variation, we stopped also to update the decision neuron's weights using the gradient-descent weight update rule. We used the entire performance evaluation and weight update stage of each epoch for measuring errors. When we tested this network, we got large oscillations of the errors that prevented or delayed the network's convergence. The reason for this was that each time the decision neuron was re-imprinted there was a chance of capturing a worse image than the one already imprinted. In order to overcome this phenomenon, it proved possible to use two decision neurons instead of one. At any given moment, only one decision neuron outputs the network's output, while the other is inactive. However, throughout the performance evaluation and weight update stage of each epoch, both decision neurons maintain error counters and at the beginning of each re-imprinting stage, the best decision neuron is fixed as the *output neuron* while the other one is re-imprinted. In this way, the decision neuron can never become worse than its present value. A problem that this solution raised was that in order for the output neuron not to be affected by the re-imprinting stage, the L2 neurons that were connected to it must not be changed. This is, however, not a significant problem, because such cases are rare due to the fact that the active neuron was connected to its 'best' children at the time and it is unlikely that they will be chosen for replacement. In the tests we performed on this variation we made sure these neurons are not replaced, although this is a non-biological step.

4.3.3.3 Biological Plausibility of the First Model Variant

Obviously, the feature selection scheme presented for this variant does not easily translate into a biologically feasible model. The most biologically unlikely part of this scheme is probably the act of 'choosing' the best and worst neurons in a column to be connected and re-imprinted. However, if one relates to the error measurements of a neuron's children as local information of the father, then this scheme does not rely on non-local information and remains within the boundaries of our restrictions. The columnar ranking procedure and the choices we made are an extreme case of a continuous and automatic mechanism. If instead of making the discrete choices, a neuron would just connect with some probability that increases as its errors decrease to its father, and would undergo re-imprinting with some probability that increases with error rate, then we may get a biologically model that resembles our feature selection choices. In this case there may be more than one neuron in a column that connects to a parent, but this should not necessarily worsen the results.

4.3.4 Model Variant II: Coping with Redundancy

The second model variant we constructed to perform object classification using the hierarchy was an improvement to the first one and an attempt to construct a biological model that resembles the algorithm described in [7]. A fundamental problem in the first model variant was that it totally ignored the amount of redundancy in its features, and did not fully exploit the potential of the fragments layer's columns for classification. We constructed a feature selection algorithm that tries to construct in each receptive field (L2 column) a set of informative features that are as little redundant as possible and that contribute to the classification together.

4.3.4.1 Variations VI and V: First Stage

As mentioned in section 4.3.2.7, our model must present a solution to the problem of multiple connections between the decision neuron and each L2 column. While the previous variant simply ignored most of the connections, in the second variant described here we tried to design a better, more complex solution. The outcome of the feature selection procedure in the first variant was that in each L2 column, the best neuron was chosen to affect the decision while the rest were unused. However, although the other neurons in the column performed worse, this does not mean that they have nothing to contribute to the decision process. Since all neurons in the same L2 column respond to the same receptive field and select features from that field, and since class objects have some variance in shape and texture, it is very reasonable to maintain a set of features that respond to the same receptive field. When classifying images of airplanes, for example, we may find out that a round cockpit is the most popular cockpit and the best feature for its part of the image, but we may also want to keep typical fragments of other, less popular, cockpits for the detection of less common airplanes. The columnar structure of L2 enables us to store such information, and the remaining problem to solve is how to combine all the neurons' outputs for the purpose of making a decision. The solution we chose to work with was to allow only one neuron to affect the decision for each presented image, and to choose for this purpose the neuron that identified its corresponding fragment in the highest degree of certainty, i.e. the neuron that showed the maximal response. Formally, we say that for the i'th L2 column, we chose the maximal output of the set of $\mathrm{NL2}=8$ neurons – $\left\{\mathrm{L2\_output}_i\left(j\right)\right\}_{j=1}^{8}$. We investigated two variations of this concept:

- **Variation IV** – A single synapse is connecting the decision neuron to each of the 36 L2 columns, with a single synaptic weight (w_i) for each one. In this case, the decision output will be calculated this way:

  $$\sum_{i=1\ldots36} w_i \cdot \max\left(\left\{\mathrm{L2\_output}_i\left(j\right)\right\}_{j=1}^{8}\right).$$

  The idea behind this variation is that each L2 column reacts to a different part of the image and therefore may be perceived as a classifier to a specific part of the class object (e.g. cockpit, tail or wheels in airplanes). Since different class images contain different types of these parts (there are, for example, a few dominating categories of cockpit shapes), it is reasonable to allow the classifier to detect a class object in any possible combination of class parts. The MAX operation in this case, will help us avoid a false-alarm caused by a weak response in a few neurons of the same L2 column.

- **Variation V** – One synapse is connecting the decision neuron to each L2 neuron, but with only one neuron per column affecting each classification result. The synaptic weight used is the one matching the selected neuron in each column. In this case, the decision output will be calculated this way:

  $$\sum_{i=1\ldots36} w_i\left(p\right)\cdot\max\left(\left\{\mathrm{L2\_output}_i\left(j\right)\right\}_{j=1}^{8}\right)$$

  while for the i'th column, $w_i\left(p\right)$ is the p'th neuron's weight:

  $$p=\arg\max_j\left(\mathrm{L2\_output}_i\left(j\right)\right).$$

  The idea here is that although each part of the class object may be important, some part-types are more likely than others, and therefore are more informative (and may have a greater weight).

In the new model variation we chose to focus on the IC fragments imprinted in L2, and to design a feature selection scheme that manages these features only, while using L1 fragments as fine-tuning for improved L2 fragments detection. This approach resembles the hierarchical, top-down construction of feature hierarchies in [7]: at first, the large, first layer fragments are chosen and only then they are broken into their building blocks for better classification results. Since we chose to focus on the IC fragments expressed by L2 features and to imitate the hierarchical FBC approach, we did not want to have L2 fragments that are composed of superpositions of different L1 fragments (taken from a few images). Instead, we decided that L2 neurons will imprint, together with a set of their children, a complete image fragment taken from a single image. As a result, we had to increase the column length of L1, NL1, to avoid L1 re-imprintings that damage existing L2 fragments. NL1 was increased, therefore, to a maximal size of 73 (at most, 72 L2 neurons can be connected to a single L1 neuron). The new neuron selection for re-imprinting process is described below.

In the initialization stage, 8 class images were presented successively. On each image, a complete set of L1 neurons (one from each L1 column) were imprinted, together with a corresponding complete set of L2 neurons. In this manner, after 8 images, we got completely initialized L1 and L2 with 'small' and 'large' fragments from 8 class images. The decision neuron's weights, connected to either all L2 columns or to all L2 neurons, were initialized with equal, constant values.

At each epoch, we used the entire performance evaluation and weight update stage (with weight update loop length of 400 images) to adjust the decision neuron's and L2 weights using the gradient-descent scheme (the learning rate we used was $\alpha = 0.05$). Error counters were updated only during the second half of this stage. Since L2 and the decision neuron's weights never ceased to change, we initialized their error counters at the beginning of every training epoch not considering whether they were affected by the last re-imprinting stage or not.

At the end of the performance evaluation and weight update stage, training class images continued to appear and be classified by the network until a miss occurred. Since this variant tries to construct L2 columns with diverse feature content, we chose to re-imprint neurons only using images that cause a miss, hoping that this will generalize the class-recognition abilities of the network. During the re-imprinting stage of each epoch, the worst L2 neuron (a single neuron) was re-imprinted together with some of its L1 children (in the same way we imprinted L1 and L2 neurons in the initialization stage) using the selected missed image. This way, we hoped to capture our weakest class part, on a problematic image. The L1 neurons chosen for re-imprinting were always free, unused neurons. Since we increased NL1 to such a size that will always allow this, there was no need in designing a neuron selection scheme for L1 neurons. At the end of each re-imprinting stage, unused L1 neurons were marked for future use. Upon re-imprinting, the decision neuron's weights did change in the variation IV. In variation V, however, the weight corresponding to the new fragment was initialized to the average of the other weights of the relevant L2 column (representing the same class-object part).

Since when we tried to use a threshold in L2 the maximal L2 output in a column was always 1 for class images, L2 threshold was discarded. L1 outputs were still restricted to positive values (without a threshold), in order to minimize the amount of negative weights and inputs in the network.

This feature selection scheme performed poorly. The reason was that it took the worst L2 neuron and re-imprinted it using a non-standard image. Therefore, that neuron always got a very high miss rate and was again and again re-chosen for re-imprinting. Moreover, since some L2

columns are essentially worse than others (some locations are more informative than others), neurons from the 'best' L2 columns were never replaced. Our goal was to achieve a variety of class fragment types, which means, for example, a set of cockpits. This should result in a column that combines some very good neurons (as the most common cockpit) with worse ones (some rare cockpits). In a way, we should expect L2 columns with large error variability (similar errors may point on similar cockpits and not on the wide 'bank' we wish to construct and rare but informative fragments may never compete with error rates of common fragments) and the scheme that always replaced the worst neuron was found inappropriate.

We tested a naïve scheme for selecting L2 neurons for re-imprinting based on their errors, in a way that would not always choose the same worst neuron. We picked from each column the neuron that had the median error rank. Doing this, we hoped to always keep the best features in each column while allowing a diversity of errors. This scheme improved the classification abilities of the network. Yet, it did not provide any means for identifying similar, redundant features. It became clear that the main problem of the learning scheme was its inability to exploit in the best way the network's limited memory.

4.3.4.2 Column Ranking Methods for Redundancy Reduction

The L2 columns' *value ranking* (in contrast to error ranking) method we constructed to replace redundant neurons was based on the relative contribution of neurons in each column. The phenomenon we tried to overcome was having a few features in the same L2 column that are very similar and that are redundant. We used the fact that the existence of such neurons can be discovered by the network's indifference to the removal of one of them, to develop the following scheme for *significance test*. Each time a 'hit' occurs (a successful classification of a class image) we test whether the elimination of the neuron with the maximal response in any L2 column would have caused a miss (each column is tested independently). Since redundant features should not cause any such change, we increase a value counter for each maximal neuron that was proved to be valuable (and caused a miss when eliminated). Similarly, we decrease the value counter of a maximal neuron whenever its elimination causes a FA to change to an 'ignore' (a successful classification of a non-class image).

At each re-imprinting stage, we chose arbitrarily a set of 9 L2 columns for re-imprinting (to minimize the chance of re-imprinting recently replaced neurons of which the weights may have not converged to achieve meaningful ranking results). In each chosen L2 column, we used the value ranks to determine which L2 neurons will be re-imprinted. The ranking scheme we constructed protected neurons with rare yet important fragments, and preferred to replace fragments that may have receive better error rankings but that are redundant.

Since the weights update scheme we used outputs in some cases negative weights, and since neurons with negative weights would never 'win' or 'lose' a value score in the ranking scheme we used (the elimination of such neurons increases the total output), we added to our significance test a complementary value test. Changing a classification result from hit to miss or from FA to ignore, involves a reduction in the total output. Therefore, opposite changes from miss to hit or from ignore to FA measure increase in the total output and match with the elimination of L2 neurons with negative decision weights. In the updated scheme, a change from miss to hit decreases the value counter; a change from ignore to FA increases it.

The ranking scheme did not improve classification results significantly, and when we investigated it we found out that the elimination of a single neuron changes the networks output only

rarely. The reason was that the difference between the outputs of the selected neuron and its replacing neuron multiplied by their relevant decision weights, rarely reaches significance. The result was that at the re-imprinting stage only very few neurons had any counts in their value counters and the replacement choices were mostly arbitrary. The solution we suggested to this problem was to use two different value credit scores that would guarantee that we have enough ranking information at the re-imprinting stage:

- A *high credit* that will increase (or decrease) the value counter by 10 whenever our significance test shows that a certain neuron was essential (or harmful) to some classification.

- A *low credit* that will increase the value counter by 1 whenever a neuron participated in a successful classification, and decrease it by 1 whenever a neuron participated in an incorrect classification (for neurons with positive decisions weights). The assumption that underlies this step is that redundant features will not evoke maximal response arbitrarily and that in most cases there will be one fragment that is more 'pure' and that will usually respond stronger to its relevant feature (and will 'win' the low credit in most cases, over redundant features). In this way, a neuron that detects a rare fragment will get a higher score than a redundant neuron that detects a common fragment but that is less efficient than another detector for the same fragment. For neurons with negative decision weights, the low credit is given in the opposite cases – reduced on a successful classification and added on a loosing one.

This updated scheme achieved much better results and clearly presented a way to eliminate some of the redundancy. In fact, only at this stage the new variations were able to compete with the first model variant. However, since the high credit was very rarely given it was practically ineffective and we lost a way to appreciate essential neurons. The last update to our significance ranking scheme managed to solve this problem and achieve the best results we got. In order for the high credit measure to be effective, we had to somehow multiply the impact of eliminating a neuron to see how important it was. Basically, when a neuron in the new model variant is eliminated, it is replaced by the neuron with the second highest output. Our problem was that the difference between these two outputs is in many cases not big enough. Since we now had the low credit method as an effective redundancy measure, we could change the high credit method to test a more global measure of the value of neuron to its entire L2 column. Instead of replacing a neuron with the second neuron in order of output strength, we replaced it with the weakest neuron and then tested if the network's output changed. This measure, which proved to be effective, achieves in a way what we initially tried to create – a way of constructing L2 columns with a large error variability. Comparing the strongest neuron with the weakest neuron in a L2 column, we actually measure the variance of that L2 column and the contribution of the extreme neurons to it.

4.3.4.3 Biological Plausibility of the Second Model Variant

The biological plausibility of the second model variant is not clear. Assuming that a biological mechanism for the ranking procedure is feasible, the feature selection method is biologically plausible in the same way that first model variant is, without the need to re-connect L2 neurons to the decision neuron. Although it is based on local information, and despite the fact that some of our mechanism's characteristics are supported by biological findings (like the MAX operation and the parallel computation pathways for measuring a neuron's essentiality), we do not describe in this work a biological system that would enable a ranking mechanism as we used in the second model variant. However, as a means to measure the redundancy of the imprinted features, our ranking procedure does have a biological rational; other redundancy measurements that were developed for

similar purposes were shown to be feasible in a biological system, and could potentially replace the mechanism we present here. One such redundancy measurement is the Merit Value (MV) that is described in [12] by Levi.

The MAX operation used for calculating L2 outputs seems compatible with neurophysiological data. For example, when two stimuli are brought into the receptive field of an Infero-Temporal (IT) neuron, that neuron's response seems dominated by the stimulus that, when presented in isolation to the cell, produces a higher firing rate ([52]). Additional support for a MAX mechanism comes from studies using a 'simplification procedure' ([53]) or 'complexity reduction' ([54]) to determine the preferred features of IT cells, that is, the stimulus components that are responsible for driving the cell. These studies commonly find a highly nonlinear tuning of IT cells which is compatible with the MAX response function. The MAX-like operation used in [8] specifically for the task of object recognition, was examined in pairs of cortical neurons of anesthetized cats in [55], and it is shown to describe, on average, action patterns of neurons in the cat's visual system.

5 Results, Conclusions and Future Work

All the tests in this work were made using test and training class images of airplanes that were fitted to the size of 33×33 pixels. Both the training and the test sets of class images consisted of 537 images each. Test and training non-class images consisted of many objects and textures and were also fitted to the same size. The training non-class images set consisted of 1450 images and the test non-class images set consisted of 1451 images. Figure 4 presents a few class and non-class images.

Figure 4: Examples for class and non-class images

5.1 Performance of Model Variant I

Figure 5 presents the performance of all three variations of the first model variant: the one that used gradient-descent weight update for L2 neurons and for the decision neuron (called here Full Gradient-Descent Model – FGDM), the one that used gradient-descent only for the decision neuron (called here Half Gradient-Descent Model – HGDM) and the one that did not use gradient-descent (called here No Gradient-Descent Model – NGDM). Performance is presented in the form of a Receiver Operating Characteristic (ROC) curve that plots the probability of a hit against that of a False Alarm (FA) for different possible decision thresholds – thresholds of the decision neuron that define the network's final binary decision.

In order to test the efficiency of the feature selection scheme, we compared the FGDM performance with those of a network that did not re-imprint any neuron and only updated L2 neurons weights and the decision neuron weights using the gradient-descent weight update rule (called here Only Gradient-Descent Model – OGDM). Surprisingly, we found out the OGDM performance were very similar and even a bit better than those of the FGDM. We also noticed that if stopped at the middle of the training process (before reaching a plateau in the error rates), the FGDM performance was sometimes better than its final performance. In order to test the work of the feature selection scheme we simulated an extreme situation where the decision neuron is affected only by the best L1

neurons and the best L2 neurons. This is the final condition that the feature selection scheme we used strives to achieve. The performance of this network was even worse than those of the original FGDM network that was used for extracting the best features in the simulation. In figure 6 it is possible to see the final performance of these 3 simulations as a ROC curve.

To make sure that our feature selection works properly, we measured the error rates of the L1 neurons that are connected to the decision neuron (via L2 neurons) at the final stage of the FGDM network and the OGDM network. It turned out that the FGDM L1 features were a little better than the OGDM features, but at the same time there were more active L1 neurons in the OGDM scheme than in the FGDM scheme. The small improvement in L1 features was not enough to compensate on the reduction in the number of L1 features. We measured the improvement in time of the FGDM scheme, and the results are presented in figure 7. This graph shows the average change in the Equal Error Rates (ERRs) of the network along the timeline of training epochs. In each training epoch, the measured ERR is the FA rate (or the miss rate) at the point on the ROC curve where these two error rates are equal. In order to speed-up the ERR measurements presented in figure 7 and to achieve a stable behavior, we stopped updating L1 neurons by re-imprinting at the 35^{th} training epoch (assuming that it should be enough for finding good enough L2 features), re-imprinted the decision neuron at each re-imprinting stage and activated the local weight update rule throughout all of the performance evaluation and weight update stage. It can be seen on the graph that together with the improvement in time due to the weight update and the feature selection schemes, there is also a slow performance deterioration that does not seem to reach convergence. We believe that this result is due to the gradual decrease in number of L1 and L2 neurons that affect the classification.

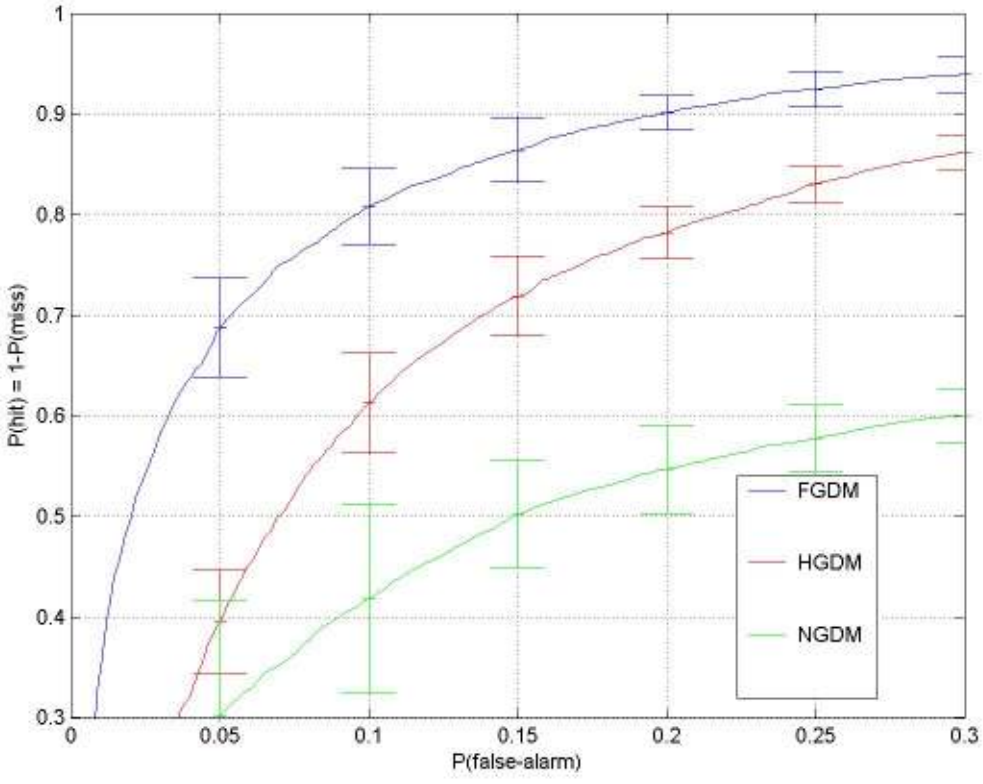

Figure 5: Performance of the FGDM, HGDM and NGDM schemes presented as a ROC curve

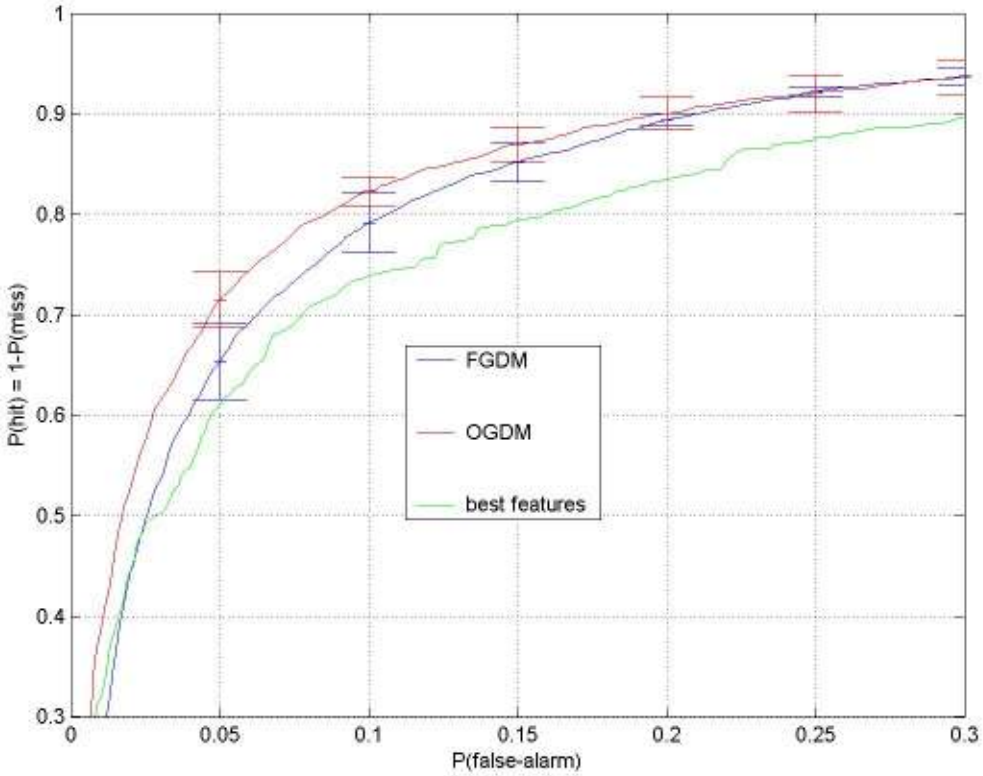

Figure 6: Performance of the FGDM and OGDM schemes compared with those of the simulated network with the best FGDM features presented as a ROC curve

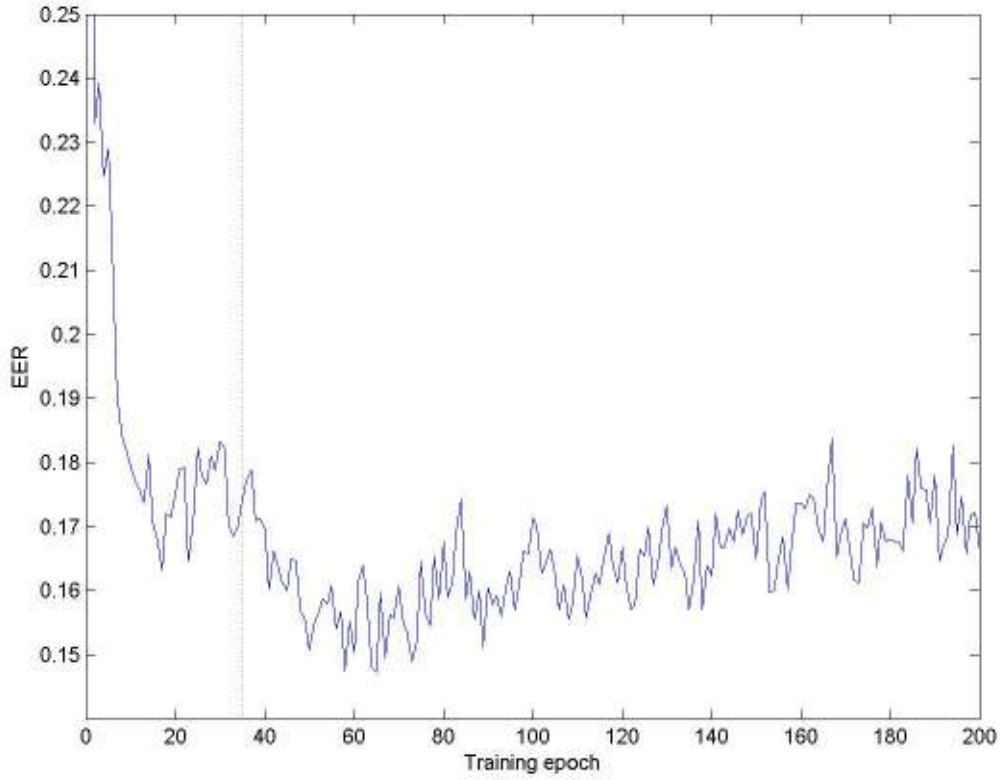

Figure 7: Change in performance of the FGDM (EERs) along the training epochs. The red dotted line marks the point where L1 stopped updating

We computed the feature fragments of the FGDM scheme by averaging the 10 images that evoked in each L2 neuron or in the decision neuron the highest responses. For L1 neurons that did not undergo neuronal weight update we chose the fragment that was imprinted. A few of the network's most important fragments (correspond to neurons with relatively high weights) are presented in figure 8. Figure 9 contains two complete L2 columns of the FGDM scheme network – columns that received (a representative from each one) the highest decision weights.

Figure 8: A sample of the FGDM scheme feature hierarchy

Figure 9: Features of two L2 columns of the FGDM scheme

5.2 Performance of Model Variant II

Figure 10 presents the performance of the two variations of the second model variant with multiple effective features in each L2 column: the one that used a single decision neuron synapse per L2 column (called here Multiple Feature Model-1 – MFM-1) and the one that used a single decision neuron synapse per L2 neuron (called here Multiple Feature Model-2 – MFM-2). Also presented in this image is the performance of the non-biological model of Epshtein and Ullman from [7]. Performance is presented in the form of a ROC curve. The non-biological model performs, as expected, better than any of the biologically restricted models.

There is no real possibility to compare these models to the first model variant, because as mentioned in section 5.1, the first model variant did not really converge and as time passes becomes less efficient. In its peak, the first model variant achieves similar performance to those of MFM-2. A possible explanation for the inability of the second model variant to outperform the first is that the MAX operation in L2 significantly harms the ability of the weight update rule to optimize results. Since in both the second variant's implementations L2 neurons fire in varying clusters, the weight update scheme is very limited compared with its action in the first model variant where the same L2 neurons are always updated together. Since as shown in section 5.1 the weight update rule has a tremendous impact on the classification performance, this characteristic of the second variant harms it.

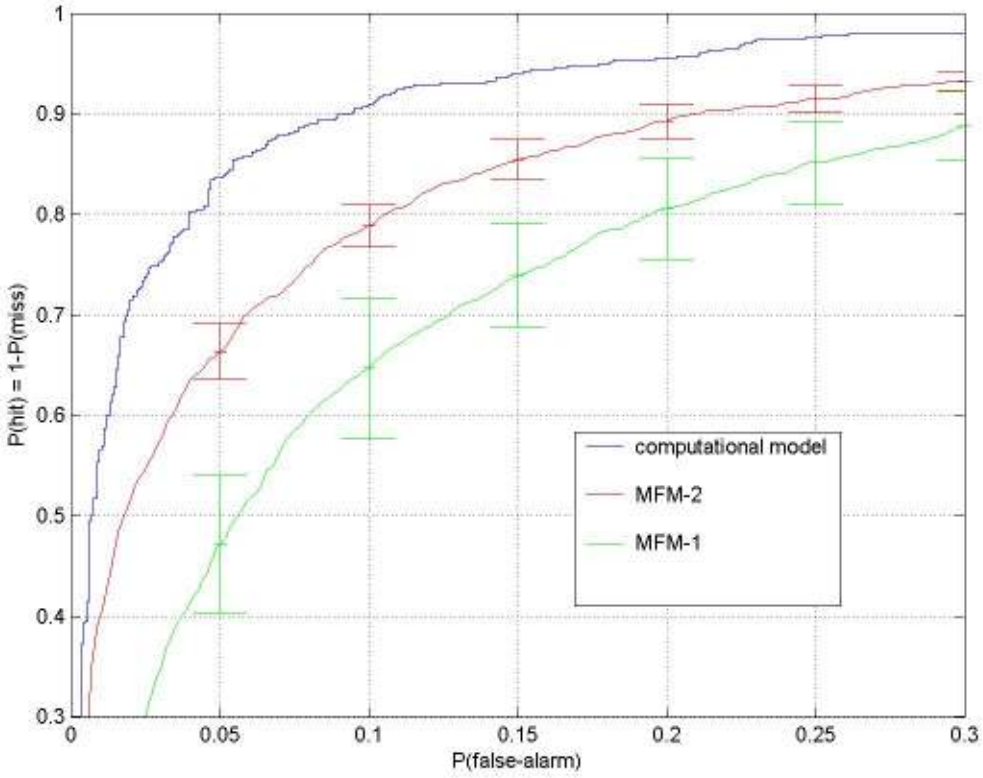

Figure 10: Performance of the MFM-1 and MFM-2 schemes presented as a ROC curve

In figure 11, a few of the MFM-2 network's most important fragments (correspond to neurons with relatively high weights) are presented. Figure 12 contains two complete L2 columns of the MFM-2 network. These are the columns that got the highest decision weights on average. The upper L2 column in this image is the exact same L2 column in the lower part of figure 9 (in section 5.1) of the FGDM scheme – it is possible to observe that the redundancy in the FGDM scheme is higher than in the MFM-2 scheme, as the variance in the MFM-2 L2 column is greater. The feature hierarchy found by the Epshtein and Ullman algorithm is presented in figure 13.

Figure 11: A sample of the MFM-2 scheme feature hierarchy

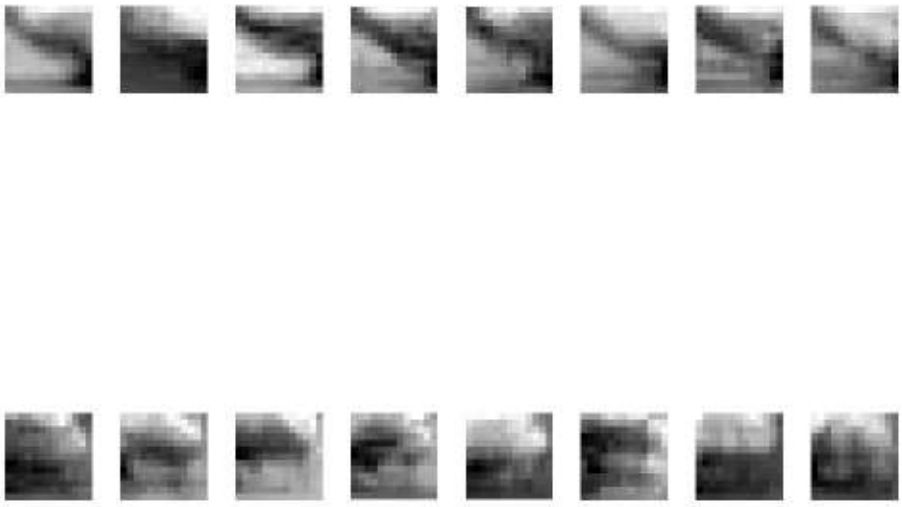

Figure 12: Features of two L2 columns of the MFM-2 scheme

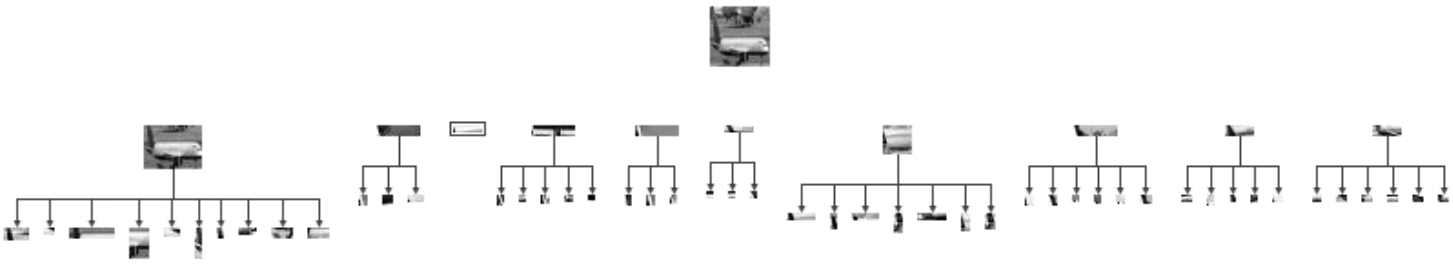

Figure 13: Feature hierarchy of the computational model of Epshtein and Ullman

5.3 Conclusions

In this work we described several attempts to construct a network model for object classification that will extract useful feature hierarchies in a neurobiologically plausible manner. We chose a set of restrictions that simulate several basic requirement of a network with biological capabilities, and managed to cope with them throughout most of the work.

A few ideas taken from different sources have been combined in this work together, to improve classification results. The mechanism of neuronal imprinting, together with a local gradient-descent weight update rule, implement an effective means of neuronal weight selection that is consistent with computational FBC concepts. The hierarchical structure of the network contributes to the classification scheme by improving the detection of Intermediate Complexity (IC) features, and the MAX operation that replaces the traditional linear computation allows a useful management of redundant features.

In addition, a novel mechanism for feature selection that uses two independent value measurements is presented in this work in a way that may be adapted to a biological system.

The three static layers that process the images prior to their transmission to the learning network simulate a real biological system and enforce more restrictions on the classification scheme. We constructed a simple network that shows similar behavior to that of real neural systems such as limited invariance to position and a form of Hebbian learning. Although the biological model's performance fall short of non-biological computational algorithm results, the simulations show that our approach to feature selection manages to decrease redundancy among the used features and to contribute to the classification scheme. When judging the results of the described classification algorithms, one should consider the fact that as a biological model we made many assumptions on different constants and fixed measures that could otherwise be fitted by computational mechanisms. In general, our goal in this project was to study and compare the efficiency of biologically plausible schemes, and not to try and compete directly with existing classification models.

Another important result of this research is that it illustrates the importance of a large set of features, but also the necessity of a mechanism that gets rid of redundant features among them. With the first model variant, experiments showed that in spite of the improvement in the usefulness of the independent features, a large set of mediocre features achieved better classification results than a smaller set of better features. Using the second model variant, we showed that a set of features selected without controlling redundancy in an explicit manner, achieves worse results than a set of the same size that is chosen while minimizing feature redundancy. A selection method that does not address the issue of redundancy may end up with a set of highly correlated features which does not produce good results. This is why, as we have observed, introducing limited randomness in the choice can produce some improved results.

We showed in our experiments that the mechanism of neuronal imprinting is not effective without some sort of a dynamic update. In a way, the neuronal weight update rule we used achieves another means of decreasing redundancy, by controlling the combined weight of redundant features that were imprinted together. The mechanism of neuronal imprinting, however, is essential due to do usefulness in classification of features which are relatively rare. Such rare-but-useful features are unlikely to evolve gradually by gradient-descent. Using the imprinting scheme, while imposing some constraints on the selection algorithm, such features may be selected. The results show that the selection scheme may run the risk of making such features candidates for re-imprinting, and avoiding the loss of such features is therefore an important requirement from the method.

5.4 Future Work

This work leaves many questions unanswered and opens a door to many possible extensions and further investigations:

- All the mechanisms that were described with just a few hints on possible directions for biologically feasible implementations may be further analyzed and broken into components which have biological relevance.

- A replacement for the gradient-descent weight update rule is needed for the fragment layer that outputs through MAX performing neurons. The update rule is designed to achieve good results after many bursts of synchronized firing of the same neurons, and in the later model this does not happen (each time a different combination of neurons fire together).

- There is place to investigate the patterns of neuron selection that are created by the MAX operation. It is reasonable to assume that for different types of class images there are clusters of L2 neurons that have high probability to fire together. If this is true, it can be a basis for an improved object classification scheme that will consider the information delivered by the combinations of L2 neurons that fire together (with maximal output in their L2 columns).

- In order for this model to be able to compete with the computational model, an extensive investigation of all the possible combinations of different parameters should be done, in order to find the best network tuning.

- It would be interesting to test the relevance of this work to real visual systems. There is already some evidence for neurons tuned to informative object fragments. A possible and interesting biological research would be to test how common these biological fragments are in real life images and see whether there are fragment detectors also for rare but unique fragments as proposed by this model.

6 References

1. Sali, E. and Ullman, S. "Combining Class-Specific Fragments for Object Classification". *Proc. Of the 10th British Machine Vision Conference (BMVC)* 1: 203-213 (1999)
2. Agarwal, S., Awan, A. and Roth, D. "Learning to Detect Objects in Images via a Sparse, Part-Based Representation". *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 26(11): 1475-1490 (2004)
3. Fergus, R., Perona, P. and Zisserman, A. "Object Class Recognition by Unsupervised Scale-Invariant Learning". *Proc. Of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 2: 264-271 (2003)
4. Heisele, B., Serre, T., Pontil, M., Vetter, T. and Poggio, T. "Categorization by Learning and Combining Object Parts". *Advances in Neural Information Processing Systems 14, Vancouver, Canada* 2: 1239-1245 (2002)
5. Leibe, B. and Schiele, B. "Interleaved Object Categorization and Segmentation". *Proc. Of the 14th British Machine Vision Conference (BMVC)* 1 (2003)
6. Ullman, S., Vidal-Naquet, M. and Sali, E. "Visual Features of Intermediate Complexity and their use in Classification". *Nature Neuroscience* 5(7): 682-687 (2002)
7. Epshtein, B. and Ullman, S. "Feature Hierarchies for Object Classification". *Proc. Of the 10th IEEE International Conference on Computer Vision (ICCV)* 1: 220-227 (2005)
8. Riesenhuber, M. and Poggio, T. "Hierarchical Models of Object Recognition in Cortex". *Nature Neuroscience* 2(11): 1019-1025 (1999)
9. Fukushima, K. "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". *Biological Cybernetics* 36(4): 193-202 (1980)
10. LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. and Jackel, L. D. "Backpropagation Applied to Handwritten Zip Code Recognition". *Neural Computation* 1(4): 541-551 (1989)
11. Rolls, E. T. and Milward, T. "A Model of Invariant Object Recognition in the Visual System: Learning Rules, Activation Functions, Lateral Inhibition, and Information-Based Performance Measures". *Neural Computation* 12(11): 2547-2572 (2000)
12. Levi, D. M. "A Neural Network Model for Fragment Based Classification". M.Sc. Thesis, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science (2004)
13. Hubel, D. H. and Wiesel, T. N. "Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex". *Journal of Physiology* (London) 160: 106-154 (1962)
14. Hubel, D. H. and Wiesel, T. N. "Receptive Fields and Functional Architecture in Two Non-Striate Visual Areas (18 and 19) of the Cat". *Journal of Neurophysiology* 18: 229-289 (1965)
15. Felleman, D. J. and Van Essen, D. C. "Distributed Hierarchical Processing in the Primate Cerebral Cortex". *Cerebral Cortex* 1(1): 1-47 (1991)
16. Hubel, D. H. and Wiesel, T. N. "Shape and arrangement of columns in cat's striate cortex". *Journal of Physiology* (London), 165:559-568 (1963)
17. Stryker, M. P., Sherk, H., Leventhal, A. G. and Hirsch, H. V. B. "Physiological Consequences for the Cat's Visual Cortex of Effectively Restricting Early Visual Experience With Oriented Contours". *Journal of Neurophysiology* 41(4): 896-909 (1978)
18. Ungerleider, L. and Haxby, J. "'What' and 'Where' in the Human Brain". *Current Opinions in Neurobiology* 4: 157-165 (1994)
19. Bruce, C., Desimone, R. and Gross, C. "Visual Properties of Neurons in a Polysensory Area in the Superior Temporal Sulcus of the Macaque". *Journal of Neurophysiology* 46: 369-384 (1981)

20. Bulthoff, H. and Edelman, S. “Psychophysical Support for a Two-Dimensional View Interpolation Theory of Object Recognition”. *Proceedings of the National Academy of Sciences of the USA (PNAS)* 89: 60-64 (1992)
21. Logothetis, N., Pauls, J., Bulthoff, H., and Poggio, T. “Shape Representation in the Inferior Temporal Cortex of Monkeys”. *Current Biology* 4: 401-414 (1994)
22. Tarr, M. “Rotating Objects to Recognize Them: A Case Study on the Role of Viewpoint Dependency in the Recognition of Three-Dimensional Objects”. *Psychonomic Bulletin and Review* 2: 55-82 (1995)
23. Booth, M. and Rolls, E. “View-Invariant Representations of Familiar Objects by Neurons in the Inferior Temporal Visual Cortex”. *Cerebral Cortex* 8: 510-523 (1998)
24. Kobatake, E., Wang, G. and Tanaka, K. “Effects of Shape-Discrimination Training on the Selectivity of Inferotemporal Cells in Adult Monkeys”. *Journal of Neurophysiology* 80: 324-330 (1998)
25. Logothetis, N., Pauls, J. and Poggio, T. “Shape Representation in the Inferior Temporal Cortex of Monkeys”. *Current Biology* 5: 552-563 (1995)
26. Perrett, D., Oram, M., Harries, M., Bevan, R., Hietanen, J., Benson, P. and Thomas, S. “Viewer-Centred and Object-Centred Coding of Heads in the Macaque Temporal Cortex”. *Experimental Brain Research* 86: 159-173 (1991)
27. Kobatake, E. and Tanaka, K. “Neuronal Selectivities to Complex Object Features in the Ventral Visual Pathway of the Macaque Cerebral Cortex”. *Journal of Neurophysiology* 71: 856-867 (1994)
28. McCulluch, W. S. and Pitts, W. “A Logical Calculus of Ideas Immanent in Nervous Activity”. *Bulletin of Mathematical Biophysics* 5: 115-133 (1943)
29. Rosenblatt, F. “Principles of Neurodynamics”. *New York: Spartan* (1962)
30. Minsky, M. L. and Papert, S. A. “Perceptrons”. *Cambridge: MIT Press* (1969)
31. Werbos, P. “Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences”. Ph.D. Thesis, Harvard University (1974)
32. Rumelhart, D. E., Hinton, G. E. and Williams R. J. “Learning Representations by Back-Propagating Errors”. *Nature* 323: 533-536 (1986)
33. Rumelhart, D. E., Hinton, G. E. and Williams R. J. “Learning Internal Representations by Error Propagation”. *Parallel Distributed Processing, vol. 1, chap. 8 (See Rumelhart, McClelland, et al.)* (1986)
34. Kohonen, T. “Self-Organized Formation of Topologically Correct Feature Maps”. *Biological Cybernetics* 43: 59-69 (1982)
35. Kohonen, T. “Self-Organization and Associative Memory (3rd Ed.)”. *Berlin: Springer-Verlag* (1989)
36. Barak, O. “Recognition by Variance: Learning Rules for Spatiotemporal Patterns”. M.Sc. Thesis, Department of Neurobiology, Weizmann Institute of Science (2005)
37. Jolivet, R., Lewis, T. J. and Gerstner, W. “Generalized Integrate-and-Fire Models of Neuronal Activity Approximate Spike Trains of a Detailed Model to a High Degree of Accuracy”. *Journal of Neurophysiology* 92: 959-976 (2004)
38. Vidal-Naquet, M. “The Extraction and Use of Informative Visual Features for Scale Invariant Recognition”. Ph.D. Thesis, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science (2005)
39. Shapley, R. and Enroth-Cugell, C. “Visual adaptation and retinal gain control”. *Osborne, N. and Chader, G. (editors), Progress in Retinal Research. Pergamon Press, Oxford*, 263–346 (1984)
40. Canny, J. “A Computational Approach to Edge Detection”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 8(6): 679-698 (1986)

41. Li, W. and Gilbert, C. D. “Global Contour Saliency and Local Colinear Interactions”. *Journal of Neurophysiology* 88(5): 2846-2856 (2002)
42. Tanaka, K. “Inferotemporal Cortex and Object Vision”. *Annual Review of Neuroscience* 19: 109-139 (1996)
43. Tanaka, K. “Columns for Complex Visual Object Features in the Inferotemporal Cortex: Clustering of Cells with Similar but Slightly Different Stimulus Selectivities”. *Cerebral Cortex* 13(1): 90-99 (2003)
44. Heeger D. “Normalization of Cell Responses in Cat Striate Cortex”. *Neuroscience* 9: 181-197 (1992)
45. Herrnstein, D. H. and Loveland C. C. “Natural Concepts in Pigeons”. *Journal of Experimental Psychology: Animal Behavior Processes* 2: 285-302 (1976)
46. Lorenz, K. “King Solomon’s Ring; New Light on Animal Ways”. *New York: Crowell* (1952)
47. Markram, H., Lubke, J., Frotscher, M. and Sakmann, B. “Regulation of Synaptic Efficacy by Coincidence of Postsynaptic Action Potentials and Excitatory Post-Synaptic Potentials”. *Science* 275: 213-215 (1997)
48. Hebb, D. O. “The Organization of Behavior”. *John Wiley: New York* (1949)
49. Senn, W., Markram, H. and Tsodyks, M. "An Algorithm for Modifying Neurotransmitter Release Probability Based on Pre- and Post-Synaptic Spike Timing". *Neural Computation* 13: 35-67 (2001)
50. Abeles, M. “Corticonics: Neural Circuits of the Cerebral Cortex”. *Cambridge University Press: Cambridge* (1991)
51. Meister, M., Wong, R. O. L., Shatz, C. J. and Baylor, D. A. “Synchronous Bursts of Action Potentials in Ganglion Cells of the Developing Mammalian Retina”. *Science* 252: 939-943 (1991)
52. Sato, T. “Interactions of Visual Stimuli in the Receptive Fields of Inferior Temporal Neurons in Awake Monkeys”. *Experimental Brain Research* 77: 23-30 (1989)
53. Wang, G., Tanufuji, M. and Tanaka, K. “Functional Architecture in Monkey Inferotemporal Cortex revealed by In Vivo Optical Imaging”. *Neuroscience Research* 32: 33-46 (1998)
54. Logothetis, N. “Object Vision and Visual Awareness”. *Current Opinions in Neurobiology* 8: 536-544 (1998)
55. Lampl, I., Ferster, D., Poggio, T. and Riesenhuber, M. “Intracellular Measurements of Spatial Integration and the MAX Operation in Complex Cells of the Cat Primary Visual Cortex”. *Journal of Neurophysiology* 92: 2704-2713 (2004)