In Search of the Minimal Recognizable Patch

Research Thesis

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Autonomous Systems and Robotics

Mark Fonaryov

Submitted to the Senate of the
Technion - Israel Institute of Technology
Sivan, 5780 Haifa June, 2020
This research thesis was done under the supervision of Professor Michael Lindenbaum in the interdepartmental program of autonomous systems and robotics.

Foremost, I would like to express my sincere gratitude to my advisor Prof. Michael Lindenbaum for the continuous support, for his patience, motivation and knowledge. His guidance helped me during the research and writing of this thesis. I would also like to thank my loving wife, Olla, for supporting and tolerating me in all times.

The generous financial help of the Irwin and Joan Jacobs Fellowship and the Technion is gratefully acknowledged.
# Contents

1 Introduction .................................................. 4

2 Object Recognition - Background ......................... 7
   2.1 Challenges of Visual Object Recognition ............... 9
   2.2 The Classification Task ................................ 9
      2.2.1 Evaluation Metrics ............................... 10
   2.3 Classical Computer-Vision Algorithms ................. 12
   2.4 Deep Learning for Computer Vision .................. 13
      2.4.1 Mathematical Notations ......................... 14
      2.4.2 Convolutional Neural Networks .................. 15
      2.4.3 Optimization and Regularization ................ 17
      2.4.4 CNNs and Partial Information ................... 19
   2.5 Human Process of Visual Recognition ................. 20
      2.5.1 Human Vision and Partial Information .......... 21

3 Tools for Determining Patch Recognizability ........... 23
   3.1 Patch Recognizability .................................. 23
   3.2 The Patch-Based Classification Model ................. 24
      3.2.1 Single Patch Network ............................. 25
      3.2.2 Patch Score Aggregation ......................... 26
      3.2.3 Placing the Softmax Layer ....................... 28
   3.3 Implementation ........................................... 29
      3.3.1 Training the PBC Model ........................... 29
      3.3.2 Regularizing the PBC Model ....................... 30
      3.3.3 Datasets ............................................ 31
      3.3.4 Main Experimental Setup .......................... 32
# 4 Patch Size and Recognizability

4.1 Single-Image Recognizability ............................................. 33

4.2 Category Recognizability ...................................................... 35

4.2.1 The Color Information Effect on Categorization ............... 36

4.2.2 The Aggregation Effect on Categorization ....................... 37

# 5 Minimal Recognizable Image-Patches

5.1 Globally Minimal Recognizable Patches .................................. 39

5.2 Locally Minimal Patches ...................................................... 40

5.2.1 Computational MIRCs .................................................... 40

5.2.2 Evaluating MIRCs with the PBC Model ............................ 43

# 6 Discussion

44

Appendices

46

A Validation of PBC Architecture and Training Process ............... 47

A.1 Categorizing Images with CIFAR10 ....................................... 48

A.2 Softmax Placement Experiments ......................................... 49

A.3 Interpolation Effect Experiment .......................................... 50

A.4 Hyper-Parameters Experiment ............................................. 50

B Informative Patch Location Experiment ................................. 52

C Occlusion Robustness Experiments ........................................ 54

C.1 Benefits of Patch-Based Classification for Occlusion Robustness 54

C.2 Regularization for Occlusion Robustness ............................. 56

C.3 Multiple Patch-Size PBC Models ......................................... 57

Bibliography .............................................................................. 61
List of Figures

1.1 Minimal patches. Top: MRPs (blue) and best smaller unrecognizable patches (red). Bottom: cMIRC examples. 5

2.1 Output examples of the four different tasks for computer vision recognition algorithms. From left: image classification, object localization, object detection, and semantic (or instance) segmentation. 8

2.2 The ten images used in a psychophysical study [Ullman et al., 2016], with their discovered MIRCs visualized by square bounding boxes. 22

2.3 Qualitative examples of human MIRCs and deep neural network FRIs. Figures from [Srivastava et al., 2019]. 22

3.1 Patch-based classification model - architecture diagram. 25

3.2 Classification difference between image-level and patch-level scores normalization. Here, there are three possible categories and two image-patches, where $S_i$ and $P_i$ are vectors of class scores and probabilities for patch $i$. With image-level normalization, scores are aggregated and the predicted class is the one associated with the highest score. With patch-level normalization, probabilities are aggregated and the predicted class, in this particular example, is the one with lowest maximal score. 28

3.3 Learning rate training schedule. 29

4.1 Confidence curves of images with the largest (left) and smallest (right) maximal drops. 34

4.2 Histograms of maximal confidence drops for classifiers trained with category-independent (left) and winner-directed (right) aggregations. 35
4.3 2D histograms of maximal confidence drop (x-axis) and its
associated patch size (y-axis), for classifiers trained with the
category-independent aggregation. .............................. 35
4.4 Mean and class-specific categorization accuracy with category-
independent aggregation. ......................................... 36
4.5 Mean and class-specific categorization accuracy of color im-
ages with category-independent aggregation. ................. 37
4.6 Mean and class-specific categorization accuracy comparison
between classifiers trained with category-independent and winner-
directed aggregations. ............................................ 38

5.1 MRP examples. Top: Images with their MRP (blue) and best
smaller unrecognizable patches (red). Middle: The MRP sub-
image. Bottom: best unrecognizable sub-image. ............... 40
5.2 Mean and standard deviation of maximal false scores (left)
and maximal correct scores (right) as functions of patch size. 41
5.3 cMIRC specification example - a patch that is q-locally recog-
nizable (blue), while all its nine contained sub-patches are not
(red). ................................................................. 42
5.4 Mean confidence drop between a cMIRC and its best contained
patch (left) and the amount of discovered cMIRCs (right) vs.
patch size. ......................................................... 42
5.5 Two images from [Ullman et al., 2016] with their human-
specified MIRCs and computationally specified cMIRCs (ex-
cluding highly overlapping patches). ............................ 43

A.1 Mean and class-specific categorization accuracy of color im-
ages from the standard CIFAR10 dataset with category-independent
aggregation. .......................................................... 48
A.2 Comparing categorization accuracy of image-level softmax and
patch-level softmax, category-independent aggregation-trained
PBC models. .......................................................... 49

B.1 Spatial consistency between most recognizable patches of dif-
cerent sizes in the CIFAR10* dataset. Overlap is calculated
relative to the smaller patch area. Average overlap area for
each patch size is stated above the x-axis. ....................... 53
C.1 Categorization accuracy of PBC models, trained on CIFAR10 with category-independent mean aggregation, as a function of the occlusion ratio. ................................. 55
C.2 Categorization accuracy of PBC models, trained on CIFAR100 with category-independent mean aggregation, as a function of the occlusion ratio. ................................. 56
C.3 Categorization accuracy of PBC-mean models, trained on CIFAR10 with different regularization techniques, as a function of the occlusion ratio. ................................. 58
C.4 Categorization accuracy of the multiple patch-size input PBC model as a function of the occlusion ratio. ................................. 59
C.5 Example of the scores histograms for images of class 1 in the test sets of CIFAR10 – OCC. The scores associated with the full-image model are red and those associated with the PCB-4 × 8 model are blue. Each column corresponds to the model’s output for a specific class and each row to the occlusion ratio of the tested images. ................................. 60
List of Tables

3.1 SPN architecture: “BN” states which layers are followed by a batch-normalization. “Same” padding refers to applying zero-padding on the input to achieve the same spatial size in the output, “Valid” padding means no-padding. ..................................... 26

3.2 Actual batch size passing through the model during training with an initial batch size of 50 images. ................................. 30

A.1 Categorization accuracy of PBC models with category-independent aggregation for different interpolated input sizes. ....... 50

A.2 Categorization accuracy of PBC models with category-independent aggregation, trained with different hyper-parameters. .... 51
Abstract

In contrast to the human recognition processes, which can rely on small and partially visible object regions to successfully recognize an object, the performance of neural networks quickly deteriorates when objects are partially occluded or cropped.

This raises a natural question: What is the minimal image part (or parts) that suffice for recognizing an object?

In this work, we consider a special practical version of this question: what is the minimal size of a square sub-image (patch) that is sufficient for recognition using a convolutional neural network? We actually consider two questions:

- **Globally minimal patch** - Here we look for a patch of minimal size that provides the correct (and best) categorization. We denote it a minimally recognizable patch, or MRP.

- **Locally minimal patch** - Here we look for a patch that suffices for correct categorization, but whose contained sub-patches do not. This criterion follows the minimally recognizable configuration (MIRC) specified in a human perception study [Ullman et al., 2016]. Here we specify it computationally and correspondingly denote it cMIRC.

To find the minimal recognizable patches, we design a special neural architecture that identifies the most informative patch and classifies the image based on the information within it.

As expected, the minimal recognizable patches we found differ between and within categories, and increase in size for higher required accuracy. A particularly surprising finding of the MIRC paper, which also motivated this study, is that human recognition accuracy drops sharply and significantly with patch size, exactly for the size separating MIRCs and their sub-patches. Interestingly, and in contrast to previous studies, we found similar sharp changes for MRPs and cMIRCs.
List of Abbreviations

ANN : Artificial Neural Network
BN : Batch Normalization
BOVW : Bag of Visual Words
BOW : Bag of Words
CIFAR : Canadian Institute for Advanced Research
CNN : Convolutional Neural Network
CV : Computer Vision
DL : Deep Learning
DNN : Deep Neural Network
DPM : Deformable Part Model
ERM : Empirical Risk Minimization
FN : False Negative
FP : False Positive
FRI : Fragile Recognition Image
GD : Gradient Descent
GT : Ground Truth
HOG : Histograms of Oriented Gradients
ILSVRC : ImageNet Large Scale Visual Recognition Competition
MIRC : Minimal Recognizable Configuration
ML : Machine Learning
MLP : Multi Layer Perceptron
MRP : Minimal Recognizable Patch
NLP : Natural Language Processing
NN : Neural Network
PA : Informative Patch Area
PAD : Informative Patch Area Distribution
PBC : Patch Based Classification
ReLU : Rectified Linear Unit
SGD : Stochastic Gradient Descent
SIFT : Scale-Invariant Feature Transform
SVM : Support Vector Machine
TN : True Negative
TP : True Positive
List of Notations

$ACC$ : Mean classification accuracy
$c$ : Class index
$E$ : Mean error rate
$F_1$ : F-score
$FAR$ : Fall-out / False alarm rate
$FN$ : Number of false negative prediction
$FP$ : Number of false positive prediction
$L_{ce}$ : Cross-entropy loss
$L_{log}$ : Logarithmic loss
$N$ : Amount of samples in dataset
$n$ : Sample index
$N_{cls}$ : Amount of classes in dataset
$N_p$ : Number of patches in an image
$P$ : Probability
$Pr$ : Precision
$Rc$ : Recall
$TN$ : Number of true negative prediction
$TP$ : Number of true positive prediction
Chapter 1

Introduction

Deep neural networks (DNNs) provide the current state-of-the-art performance in many computer vision tasks, and especially in recognition [Krizhevsky et al., 2012, Girshick et al., 2014, He et al., 2016]. In contrast to human recognition processes, which can successfully recognize small or only partially visible objects [Tang et al., 2018, Ullman et al., 2016], the performance of neural networks quickly deteriorates when objects are partially occluded or cropped [Pepik et al., 2015, Osherov and Lindenbaum, 2017].

This raises a natural question: What is the minimal image part (or parts) that suffice for recognizing an object?

In this work, we consider a special practical version of this question: what is the minimal size of a square sub-image (patch) that is sufficient for recognition using a convolutional neural network (CNN)? We chose to consider a CNN as a substitute to general recognizability because, currently, CNN algorithms are as good as, if not better than, any other algorithm at this task. We actually examine that question for two types of patches:

**Globally minimal patch** - This is a patch of minimal size that provides the correct (and best) categorization. We denote this sub-image the minimally recognizable patch, or MRP.

**Locally minimal patch** - This is a patch that suffices for correct categorization, but whose contained sub-patches, do not. This criterion follows the *minimally recognizable configuration* (MIRC) specified in [Ullman et al., 2016] using human responses (see Section 2.5.1). In this paper, we specify this patch computationally and correspondingly denote it cMIRC.
Figure 1.1: Minimal patches. Top: MRPs (blue) and best smaller unrecognizable patches (red). Bottom: cMIRC examples.

Note that cMIRCs are not unique and are also not of minimal size. The MRP on the other hand, is unique by definition and is of minimal size, no larger than the size of the smallest cMIRC in the given image (see examples in Figure 1.1). Being minimal, it may be considered as one way for quantifying the minimal amount of information required for recognition.

To find minimal recognizable patches, we designed a special neural architecture that identifies the most informative patch and classifies the image based on the information within it. Several variations of this patch-based classification (PBC) architecture, corresponding to different patch sizes and different ways of accumulating the local information, are considered. As expected, the minimal recognizable patches we found differ between and within categories, and increase in size for higher required accuracy.

A particularly surprising finding of the MIRC paper [Ullman et al., 2016], which motivated this study, is that human recognition accuracy drops sharply and significantly with patch size, exactly for the size separating MIRCs and their sub-patches. This is in contrast to algorithmic classifiers where the accuracy decreases smoothly with patch size, and suggests that a different mechanism is applied in the human system. Interestingly, and in contrast to previous studies, we found similar sharp changes for MRPs and cMIRCs.

This study offers the following contributions:

- PBC neural architecture that finds the most informative patch and uses it to categorize the entire image.
- Characterization of globally and locally minimal patches sufficient for categorization.
- An investigation of the confidence vs. patch size behavior, pointing at similarities between patch-based categorization in human vision and in algorithmic implementations.
The remainder of this thesis is organized as follows. In Chapter 2, we begin with a summary of the required background for this work, including a review of object recognition tasks and classical and CNN-based algorithms used for computer vision. We also discuss several related works that evaluate the amount of information required for human and algorithmic recognition. We continue in Chapter 3 with a thorough introduction and analysis of the PBC model, the main tool we developed for evaluating minimal recognizable patches. Chapters 4 and 5 describe the principal experiments conducted in search for the minimal recognizable patches, including results and comparison with a related human vision study [Ullman et al., 2016]. We conclude our work and discuss its results in Chapter 6. Several additional experiments that were performed as part of this study are provided in the appendix.

Some results in this thesis were submitted to the ECCV2020 conference.
Chapter 2

Object Recognition -
Background

Object recognition is the process of combining previously accumulated knowledge with newly available visual information for the purpose of meaningful observation and distinction. For humans, it is a primal and vital ability, required for many day-to-day tasks, for example determining between familiar and unfamiliar faces, or separating edible and non-edible substances. It is also a classical task in the field of computer vision (CV). The process of object recognition is comprised of two different tasks:

- **Object categorization** - assigning an object to a certain category (a.k.a. generic object recognition), e.g., separating between images of cats and dogs.

- **Object identification** - recognizing a particular, distinctive instance of an object (a.k.a. specific object recognition), e.g., pointing out a familiar friend from a crowd.

The study of visual object recognition addresses the problem of recognizing three-dimensional (3D) objects given only their two-dimensional (2D) projections. This holds true both for humans receiving 2D patterns of light on their retina, and for computer-vision algorithms operating on images or video.

In the field of computer vision, it is common to distinguish between four recognition tasks:

- **Image classification** - Given an input image, assign it a class label. This particular task is further explained in Section 2.2.
• Object localization - Given an input image and its class label, find the exact location of the object, i.e., draw a bounding box around it.

• Object detection - A combination of the previous two tasks: Given an input image, draw a bounding box around every object of interest in the image and assign them a class label. The output is one or more bounding boxes, each with its own class label.

• Semantic segmentation - The process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics. The output is an array with the same spatial shape as the input image, where every pixel contains the class label information.

An example of the required output for every recognition task is presented in Figure 2.1.

In the next sections of this chapter, we examine several aspects of object recognition. We first discuss what makes object recognition an interesting but difficult challenge in Section 2.1. We then elaborate on the task of image classification in Section 2.2. We briefly review several classic algorithms designed for object recognition in Section 2.3, and then provide a more thorough review of deep learning algorithms, including the mathematical formulation used in neural networks, in Section 2.4. We follow with a discussion about the human process of visual recognition and a survey of works related to our study in Section 2.5.

Figure 2.1: Output examples of the four different tasks for computer vision recognition algorithms. From left: image classification, object localization, object detection, and semantic (or instance) segmentation.
2.1 Challenges of Visual Object Recognition

Visual object recognition is a challenging computational problem. A three-dimensional object can be projected to a large number of different two-dimensional images, and the variability of its appearance is affected by both external and internal variables.

Even a single rigid part, can be viewed from many different angles, viewing distances, and lighting conditions, correspondingly changing the object’s projection orientation, scale, and brightness distribution. With these external variables, we can also include different backgrounds, which influence the ability to separate an object from its surrounding.

In addition, internal variability also has a considerable contribution. Most objects are comprised of more than a single part, so even if each part is by itself rigid, the overall structure can have many different pose configurations.

As a result, even for the task of specific object recognition, where an object’s variability is relatively constrained, there are still numerous possible 2D projections. For generic object recognition, within-class variability adds an additional layer of computational complexity. For example, there are many types of dogs, each notably different, but all must be clustered to a single category.

2.2 The Classification Task

The fundamental classification problem is binary classification, where only two possible groups are available and the task is to predict the correct membership of new instances, on the basis of a specific classification rule. Adding additional groups transforms the problem into multi-class classification [Alpaydin, 2014].

In both binary and multi-class classification problems, each data instance can only get a single label. Multi-label classification [Tsoumakas and Katakis, 2007] is a generalization of these tasks, where there is no limitation on the number of assigned labels. Formally, classification is defined as the problem of using prior knowledge, based on a training dataset containing samples of known classes, to estimate and allocate labels for new instances [Alpaydin, 2014].

The vast majority of classification algorithms in computer vision are designed and evaluated in a closed set setting, where all test classes are known
a-priori [Scheirer et al., 2013]. A more challenging, yet more natural scenario for vision applications is open set recognition, where during training, only partial knowledge of the world is present, and unknown classes may be encountered during evaluation. [Scheirer et al., 2014] divided all possible classes that a classification algorithm may face into three basic types:

- **Known positive classes** - Classes for which labeled instances are available in the training set.
- **Known negative classes** - Classes for which instances are included in the training dataset as negative examples for other classes. The negative class’s label may be present or not.
- **Unknown negative classes** - Classes that are not encountered in the training process.

### 2.2.1 Evaluation Metrics

For classification tasks, there are four terms used to describe the relationships between the classifier output for specific instances and their actual ground-truth:

- **True positive** - Correctly predict the positive class, e.g., an image is correctly classified.
- **True negative** - Correctly predict the negative class, e.g., an image is not classified as the wrong class.
- **False positive** - Incorrectly predict the positive class, e.g., an image is classified as the wrong class.
- **False negative** - Incorrectly predict the negative class, e.g., an image is not classified as the correct class.

Based on these terms, we can define the metrics required for evaluating classification algorithms on a certain dataset. The first most common and quintessential classification metric is accuracy ($ACC$), defined in Equation 2.1 as the fraction of correct predictions out of the total number of samples in the dataset.

$$ACC = \frac{TP + TN}{TP + FP + FN + TN} \quad (2.1)$$
Where \([TP, TN, FP, FN]\) stand for the amount of instances in the evaluated dataset per term.

The error rate \((E)\) is the complement metric of accuracy. It is defined in Equation 2.2 as the fraction of wrong predictions within the total number of samples in the dataset.

\[
E = \frac{FP + FN}{TP + FP + FN + TN} = 1 - ACC \tag{2.2}
\]

There are many cases in which accuracy is not enough to describe the performance of a classifier, especially if the dataset is imbalanced (i.e., a disproportionate ratio of observations in each class). Two additional commonly used metrics are precision \((Pr)\) and recall \((Rc)\), both of whom are measures of relevance. Precision is used to describe the fraction of predicted positives out of true positives, while recall is the fraction of true positives, correctly classified; see Equations 2.3-2.4. In simple terms, high precision means a classifier’s output has more relevant results than irrelevant ones, while high recall means that it includes most of the relevant results in the dataset.

\[
Pr = \frac{TP}{TP + FP} \tag{2.3}
\]

\[
Rc = \frac{TP}{TP + FN} \tag{2.4}
\]

To convey the balance between recall and precision in a single variable, their harmonic mean is calculated under the name F-score \((F_1)\); see Equation 2.5.

\[
F_1 = \frac{2 \cdot Pr \cdot Rc}{Pr + Rc} \tag{2.5}
\]

One additional metric employed in our study is fall-out, also known as false alarm rate \((FAR)\), defined in Equation 2.6. It is a measure for the probability of false detection.

\[
FAR = \frac{FP}{TN + FP} \tag{2.6}
\]

All aforementioned metrics only consider the classification output of the classifier, ignoring its confidence in said output. Logarithmic loss \((L_{\text{log}})\) is different, and considers both classification correctness and its probability. Therefore, in addition to evaluation, \(L_{\text{log}}\) is used as an optimization objective.
during training for tasks of binary classification; see 2.7. Its generalization for the multi-class problem, called categorical cross-entropy ($L_{ce}$), is defined in Equation 2.8.

$$L_{log} = -\frac{1}{N} \sum_{n=1}^{N} \left[ y_n \log(\hat{y}_n) + (1 - y_n) \log(1 - \hat{y}_n) \right]$$

(2.7)

Where $N$ is the number of samples in the dataset, $y_n \in [0, 1]$ is the ground truth label of sample $n$ and $\hat{y}_n \in [0, 1]$ is the classifier’s output probability.

$$L_{ce} = -\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{N_{cls}} y_{nc} \cdot \log(\hat{y}_{nc})$$

(2.8)

Where $N_{cls}$ is the number of classes in the dataset, $y_{nc} \in [0, 1]$ is the ground truth label of sample $n$ for class $c$ and $\hat{y}_{nc} \in [0, 1]$ is the classifier’s output probability for said class.

### 2.3 Classical Computer-Vision Algorithms

Prior to the rise in popularity of deep neural networks, computer vision algorithms relied on feature engineering and classical machine learning. Features are crucial for predictive models and greatly affect their results. Thus, a considerable portion of research was focused on inventing new and improved visual features. Important examples of engineered features include wavelets [Mohan et al., 2001], scale-invariant feature transform (SIFT) descriptors [Lowe, 2004], edge orientation histograms [Freeman and Roth, 1995], and histograms of oriented gradients (HOG) [Dalal and Triggs, 2005].

Local regions and features have been widely used by classical recognition algorithms and have provided improved immunity to pose change, partial occlusion, and in-class variance. Prominent examples include image fragments [Vidal-Naquet and Ullman, 2003], visual bag-of-words models [Csurka et al., 2004], and the deformable parts models [Felzenszwalb et al., 2008].

The BOVW algorithm originates from a natural language processing (NLP) oriented approach, named the bag of words (BOW) [Manning et al., 1999]. One of the main issues in NLP is that text can come in many different lengths, which is a problem for classification models, so it needs to be quantized. The general idea of BOW is to divide a text of any size into individual
words and associate them with a known vocabulary of words (features), ignoring grammar or order and keeping only a histogram of feature appearances. This simplified representation, commonly relying on the WordNet database [Miller, 1995] for feature representation, led to many improvements in text classification [Scott and Matwin, 1998]. The bag of visual words treats local image features in a similar manner to words in BOW. It ignores their relative position, scale, and orientation, and quantizes an image into a sparse histogram of feature appearances. The simplified representation can then be used to train a discriminative classifier, such as a support vector machine (SVM). For example, [Csurka et al., 2004] used SIFT features as a basis for a visual vocabulary and trained a BOVW classifier. [Lazebnik et al., 2006] offered an extension to the orderless BOVW approach by partitioning the image into increasingly fine sub-areas and computing histograms of local features for each independently. The resulting “spatial pyramid” representation provides the classifier with basic geometrical relations between features.

The DPM algorithm for object detection, proposed by [Felzenszwalb and Huttenlocher, 2005], relies on local features as well, but in contrast to the bag of words, it also considers their spatial positions. In DPM, objects are represented as a collection of parts, ordered in a deformable configuration. Each part is a local visual feature, while the deformable configuration is characterized by spring-like connections between established (predefined or learned) pairs of parts [Felzenszwalb et al., 2010].

2.4 Deep Learning for Computer Vision

Deep learning is the current common approach for most machine learning problems, in a variety of application domains. Computer vision in particular has made considerable gains since the recent rise in deep learning algorithms, with deep convolutional neural networks (CNN) providing the state-of-the-art performance for practically every task.

Deep learning models are essentially large and deep artificial neural networks (ANN). An ANN can be well presented as a directed acyclic graph, consisting of multiple stacked layers, each with a large number of simple elements called neurons. The combined product of neurons from layer $x$ acts as the input feature vector for layer $x + 1$. A complete network consists of repeating this process over multiple layers, until reaching the final output. Each artificial neuron incorporates a two-step operation; it first calculates
the weighted sum over all input features and then this intermediate output is passed through an activation function to obtain the neuron’s final output.

The activation function is required to be both nonlinear and differentiable. Non-linearity allows the network to learn complex and nonlinear functions over the input data. Otherwise, any number of layers can be collapsed into a single linear operation. Differentiability is needed for the backpropagation algorithm when training to optimize network’s weights. Conventional activation functions include the sigmoid, hyperbolic-tangent, and rectified linear units (ReLU). See Section 2.4.1 for a formal definition.

ANNs are not a new idea. In fact, the perceptron model was first introduced in [Rosenblatt, 1958] as far back as 1958, and the idea for the backpropagation algorithm, used to update the network parameters via gradient descent, was proposed in the context of control theory in 1960 [Kelley, 1960]. Even the vision-oriented CNN class of deep neural networks was proposed by Yann LeCun more than 30 years ago [LeCun et al., 1989]. Yet, the current deep trend is only several years old, starting from the AlexNet [Krizhevsky et al., 2012] win (by a large margin) of the ImageNet large scale visual recognition challenge (ILSVRC) in 2012.

2.4.1 Mathematical Notations

Throughout this study, we refer to one-dimensional arrays as vectors, two-dimensional arrays as matrices, and any array of higher dimensionality as tensors. The zero-dimensional elements, comprising each array, are denoted with lower case letters (e.g., \( a, a_i \)), while vectors are represented with bold lower case letters (e.g., \( \mathbf{v} \)). Matrices are indicated with ordinary bold upper case letters (e.g., \( \mathbf{M} \)), while calligraphic bold upper case letters symbolize tensors (e.g., \( \mathcal{T} \)).

Let \( n_{d}^{[l]} \) be the notation for a single neuron \( d \in [1, D_l] \) at layer \( l \in [1, L] \), where \( D_l \) is the number of individual neurons in layer \( l \) (a.k.a. its width), and \( L \) is the total amount of layers in the network. Each neuron \( n_{d}^{[l]} \) takes in an input vector of size \( n_{x_l} \), and produces a single output:

\[
y_d = \phi \left( \sum_{i=1}^{n_{x_l}} w_i \cdot x_i + b \right) = \phi(z_d)
\]

Where \( w_i, b_i \) are the trainable weights and bias parameters of the neuron, \( \phi \) is the activation function and \( z_d \) is its intermediate weighted sum output.
The weighted sum operation can also be written in vector form:

\[ z_d = w^T \cdot x + b \]  
(2.10)

Where \( w, x \) are \( n_{x_l} \)-dimensional column vectors. This notation allows us to write the operation for the entire layer \( l \) as follows:

\[ y^{[l]} = \Phi \left( W^{[l]} x + b^{[l]} \right) \]  
(2.11)

\( W^{[l]} \) is a weights matrix of \( n_{x_l} \) columns and \( D_l \) rows, and \( b^{[l]} \) is a bias column vector with \( D_l \) elements. The \( \Phi \) denotes a row-wise activation function.

There are many options for the activation operation, however the three most common, as well as those we use in this study, are the sigmoid, ReLU, and softmax functions. The ReLU and sigmoid functions are defined in the following equations:

\[ \sigma(z) = \text{sigmoid}(z) = \frac{1}{1 + e^{-z}} \]  
(2.12)

\[ R(z) = \text{ReLU}(z) = \max(0, z) \]  
(2.13)

Out of these two, the sigmoid function (or its equivalent hyperbolic tangent) appears to be more biologically plausible. However, using a ReLU activation results in faster training convergence, in addition to a slightly better outcome, in part due to its ability to produce a sparse output. The softmax activation function is usually only applied as the final activation of a multi-class classification network, since it takes as input a vector of \( k \) real numbers, and normalizes it into a probability distribution consisting of \( k \) probabilities proportional to the input’s exponentials.

\[ SM(z_i) = \text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{i=1}^{k} e^{z_i}} \]  
(2.14)

### 2.4.2 Convolutional Neural Networks

Convolutional neural networks are a specialized class of ANNs, aimed to work with input data that contains a grid-like topology. This makes them specifically suited for vision tasks, since digital images are essentially represented on a 2D grid of pixels. A CNN exploits the spatial locality of image features and replaces the general matrix multiplication, performed in standard
ANN layers (shown in Equation 2.11), with a specialized kind of linear operation, named “convolution”. Layers that apply a convolution operation are called convolutional layers, while layers with a general matrix multiplication operation are called dense or fully-connected layers.

Formally, a discrete convolution operation over a 1D input $x$ with a weighting function $w$ is defined in the following equation:

$$s(t) = (x * t)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t - a) \quad (2.15)$$

For a finite two dimensional input $I$ of size $(X,Y)$ and a two-dimensional kernel $K$ of size $(M,N)$, the 2D convolution operation for spatial position $(x,y)$ is defined as:

$$s(x,y) = (I * K)(x,y) = \sum_{m=1}^{M} \sum_{n=1}^{N} I(m,n)K(x-m,y-n) \quad (2.16)$$

Equation 2.16 is only valid if the input contains only a single channel of information, as in the case of gray level images. In most cases, an input image contains three channels (RGB), and every following feature map has an even larger number. Yet, the local dependency limitation is usually applied only to the two spatial dimensions, while being computed over all channels of the input.

$$s(x,y,c) = (I * K)(x,y,c) = \sum_{m=1}^{M} \sum_{n=1}^{N} \sum_{c=1}^{C} I(m,n,c)K(x-m,y-n,c) \quad (2.17)$$

Equation 2.17 is the convolutional equivalent of a single neuron and produces a single feature. This type of model structure relies on the locality of visual features and also contains an inbuilt invariance to big translations (larger than kernel size).

While convolutional layers are invariant to big translations, they have an issue with local variations in the input. A convolution operation records the precise position of input features, thus minor shifts result in a different feature map. The common approach to this issue is down-sampling the original signal. Two regularly used pooling methods are average-pooling and max-pooling, which summarize either the average or most activated presence.
2.4.3 Optimization and Regularization

The goal of a machine learning algorithm is to reduce the true risk of performing its prediction. However, for most cases, the true distribution of data is unavailable, thus the true assessment of an algorithm’s ability is impossible. Instead, its performance is evaluated on a known set of training data. Optimization of a model over a training set can be performed by either supervised learning, if said dataset includes samples of true input-output pairs, or via unsupervised learning, if it only contains input data without labeled responses. Training with partial information is referred to as semi-supervised learning.

Supervised learning requires us to define a loss function (i.e., empirical risk), which maps a model’s output onto a real number, with the goal of minimizing it. For multi-class classification, a common approach is to use a categorical cross-entropy loss, defined in Equation 2.8.

During supervised training, the model’s parameters are updated via some form of gradient descent (GD), which is a first-order iterative optimization algorithm. The original GD algorithm would calculate a gradient for the entire dataset (i.e., full batch) for a single iteration. Another approach is to update the weights after every single sample, picked in random order, which on average would achieve the right gradient. However this process, known as stochastic gradient descent (SGD), results in a very noisy learning process. The common practice is to update weights after a mini-batch of several samples. Formally, this process is called mini-batch SGD, yet it is commonly referred to as SGD, with batch and mini-batch being used interchangeably, with both meanings implying the original mini-batch definition. We also follow this commonly used phrasing practice.

A single weights update step with SGD optimization is defined in Equation 2.18

$$w_{t+1} = w_t - \eta \nabla L(w_t)$$

(2.18)

where $\eta$ is the learning rate, used to set the scale of the weights update, and $\nabla L(w_t)$ is the loss function’s derivative, calculated with the weights at step $t$. There are several variations to this optimizer, and in our study we use one
named adaptive moment estimation (Adam), defined in Equation 2.19

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{\nu}_t + \epsilon}} \hat{m}_t$$

(2.19)

where $\hat{m}_t$ and $\hat{\nu}_t$ are the bias-corrected estimates of the first and second moment of the gradients, $\beta_1, \beta_2$ are their decay rates, and $\epsilon$ is a small smoothing term, required for numeric stability. See Equations 2.20a and 2.20b for a formal definition.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1}, \quad m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla L(w_t)$$

(2.20a)

$$\hat{\nu}_t = \frac{\nu_t}{1 - \beta_2}, \quad \nu_t = \beta_2 \nu_{t-1} + (1 - \beta_2)((\nabla L(w_t))^2$$

(2.20b)

The Adam optimizer, proposed in [Kingma and Ba, 2015], is a combination of two earlier algorithms, called RMSprop and SGD with momentum [Tieleman and Hinton, 2012, Qian, 1999]. Adam uses the squared gradients to scale the learning rate for individual parameters like RMSprop, and also takes advantage of momentum by using a moving average of the gradient instead of gradient itself, similar to SGD with momentum. See [Ruder, 2016] for a complete overview of different optimization techniques.

When an algorithm performs well on the training set but poorly on the testing set, the algorithm is said to be overfitting. In simple terms, the algorithm becomes too complex and relies on insignificant trends in the training data. Due to the vast number of parameters used in DNNs, overfitting becomes a major challenge. [Zhang et al., 2017] showed that a standard CNN has enough representative capacity to find a perfect fit to a dataset with random labels.

This issue is tackled by various regularization techniques, aimed to simplify the model. Two simple and effective methods are the ridge and lasso regularizations, which respectively add $L_2, L_1$ penalties on a model’s weights to the loss function.

Dropout regularization, proposed in [Srivastava et al., 2014], is an efficient approach that approximates training a large number of neural networks with different architectures in parallel. During training, a predetermined portion of a layer’s outputs are randomly set to zero. In effect, each weight update, for a specific layer, is performed based on different input features, which helps it break unnecessary connections in the input.
Batch normalization [Ioffe and Szegedy, 2015] is a technique designed for improving training speed, which also contains an inherent regularization effect. Put simply, each batch is standardized during training to have a mean of 0 and variance of 1. This helps to reduce the covariance shift of input features.

Another option is to artificially increase the training set size by using data augmentation. Common image augmentation techniques include random flipping, cropping, zooming, shifting, and brightness adjustment. See [Shorten and Khoshgoftaar, 2019] for additional information.

### 2.4.4 CNNs and Partial Information

The process carried out by CNNs of combining local features with increasingly more global and expressive structural information provides the current state-of-the-art performance in classification and detection of fully visible objects. However, the recognition accuracy of CNNs decreases when faced with partially visible objects, as happens for example in the presence of occlusions [Pepik et al., 2015, Opitz et al., 2016, Osherov and Lindenbaum, 2017].

A CNN’s robustness to partial occlusions can be improved by incorporating samples of occluded objects during training [Pepik et al., 2015]. However, not every possible occlusion can be encountered in the training dataset as the occlusion distribution seems to follow a long tail [Wang et al., 2017]. In addition, the effects of such an approach are limited and do not generalize well [Geirhos et al., 2018]. Specialized dedicated modifications to standard recognition architectures either exploit domain-specific prior knowledge [Zhang et al., 2018] or work by reducing the spatial support of their learned features [Osherov and Lindenbaum, 2017]. These approaches improve their partial object accuracy, which however is still considerably lower than that obtained with full-object visibility.

This deterioration is expected, yet it stands in contrast to human vision, in which object recognition is attained remarkably well, even when seeing only partial data.

The amount of information required for algorithmic recognition was considered in several works. Image regions of intermediate complexity were found to be maximally informative [Ullman et al., 2002]. Sensitivity of common CNN architectures to occlusions was evaluated in various studies [Pepik et al., 2015, Geirhos et al., 2018, Tang et al., 2018]. Robustness to different image distortions such as blur, noise, contrast and JPEG compression...
was also examined [Dodge and Karam, 2016, Geirhos et al., 2018]. It seems that local features carry a lot of information and suffice for high accuracy classification [Brendel and Bethge, 2019].

2.5 Human Process of Visual Recognition

The human vision system is far from being fully understood, yet it is clear that humans can recognize objects effortlessly and with tremendous speed [Thorpe et al., 1996], and use limited or partial data sufficiently well for high accuracy [Torralba et al., 2008, Ullman et al., 2016].

Classical recognition models hypothesize that visual recognition entails several stages, where at first the presence of an object is detected, and only then it is categorized and later identified at a finer grain. An intermediate stage between low-level visual processing and high-level object recognition, at which the object’s location within the scene is first detected before it is further processed for recognition, was included in many early models [Bregman, 1981, Nakayama et al., 1995, Driver and Baylis, 1996]. The intuition was that an efficient recognition process will not perform high-level computations indiscriminately at every location of an image (or field of view), since a majority of regions will not contain any objects of interest.

However, there is also evidence to suggest that object recognition may precede general object detection (or segmentation), and in fact might be used to affect results of the detection process [Peterson and Gibson, 1993, Peterson and Gibson, 1994, Peterson and Kim, 2001]. Some experiments in [Grill-Spector and Kanwisher, 2005] also revealed that the processes of localization and categorization are intertwined to the point of indifference, suggesting that they might be a single process.

Visual information is used by integrating it with prior knowledge that has been accumulated about the world [Ward, 2015]. Object constancy, meaning the ability to recognize an object across different viewpoints and lighting conditions, is an important aspect of object recognition. A common thesis is that object constancy is achieved by matching the constructed visual representation with a cache of memorized object descriptions that carry information about invariant properties of objects. In other words, because comparing low-level visual information is inefficient, matching needs to be performed on a higher level of features.

The question of how objects are stored and represented in the human
mind also remains open. An early suggestion was that the brain stores only structural descriptors of the typical views, and any atypical viewing point requires a process of transformation [Marr and Nishihara, 1978, Tarr and Pinker, 1989], which results in longer recognition time [Palmer, 1981]. Other works argue that object constancy is achieved by matching a reconstructed three-dimensional representation of the viewed object with a learned structural descriptor [Biederman, 1987, Hummel and Biederman, 1992]. In general, approaches to neural object representation can be divided into the viewpoint-invariant approach suggested by [Biederman, 1987], and the viewpoint-dependent approach applied by [Tarr and Bülthoff, 1995].

2.5.1 Human Vision and Partial Information

While the human object recognition process remains debatable, it clearly works well with partial data. First, low resolution is sufficient [Torralba et al., 2008]. Human scene recognition was evaluated for several image resolutions ($8^2, 16^2, 32^2, 64^2, 256^2$), and $32 \times 32$ color (or $64 \times 64$ gray-level) images are recognized well. Second, human object recognition abilities remain robust when only small amounts of information are available due to heavy occlusion; even 10% visibility is adequate for performance well above chance [Tang et al., 2018]. Interestingly, partially visible objects require longer time for recognition, which hints to a possibly different process than recognition of fully-visible objects [Tang et al., 2018].

Recently, a psychophysical study [Ullman et al., 2016] further showed that reliable human object recognition is possible even from small image-patches and specified a special class of minimal image patches. These patches are minimal in the sense that they are recognizable, but their sub-patches, smaller by 20%, or identical patches with 20% lower resolution, are not. That is, such patches, denoted minimal recognizable configurations (MIRCs), are locally minimal (see Figure 2.2). Interestingly, this study found that human recognition accuracy associated with the sub-patches was significantly lower than that associated with the MIRC itself. Tests with recognition algorithms, applied to MIRCs and to their sub-images, did not find a similar accuracy drop. A follow-up on this work demonstrated that CNN classification for some patches, denoted fragile recognition images (FRIs), may be changed due to small translation or to small resolution reduction [Srivastava et al., 2019]. Found FRIs are different from MIRCs in several aspects, including size, location, frequency, and sharpness of the drop in recognition rate (see
Figure 2.3). Our work relates to the psychophysical study mentioned above [Ullman et al., 2016] as we use our model to both evaluate the MIRCs and to suggest computationally specified locally minimal recognizable patches, denoted cMIRCs, which exhibit comparable properties to the human-specified MIRCs. Similarly to FRI patches in [Srivastava et al., 2019], cMIRCs are locally minimal patches that are specified by a computational model. However, unlike an FRI, which is defined by a change in predicted classification output, cMIRCs are specified based on the classifier’s prediction confidence. We also find another type of globally minimal patches, denoted MRPs, and observe that both types of minimal patches in our study are characterized by sharp drops in recognition.

Figure 2.2: The ten images used in a psychophysical study [Ullman et al., 2016], with their discovered MIRCs visualized by square bounding boxes.

Figure 2.3: Qualitative examples of human MIRCs and deep neural network FRIs. Figures from [Srivastava et al., 2019].
Chapter 3

Tools for Determining Patch Recognizability

3.1 Patch Recognizability

Our goal is to characterize the globally and locally minimal sub-image (patch) required for successful categorization. In this study, we consider this general question in the context of a specific data set and in the closed set setting.

To address image variation, due to scaling and other pose changes, we specify patch size as a fraction of the full object size, as seen in the given image. In our study, we use fixed-size images (32 × 32) from the CIFAR datasets. Most images contain one object, tightly bounded by the image boundaries. For these images, a fixed fraction of the object size corresponds to a fixed size in pixels.

Consider a specific patch. We shall denote this patch as locally recognizable if a categorization procedure accepting only this patch as its input classifies it to the correct category. Formally, let $S^c_p$ denote the score of class $c$ associated with the patch $p$. Here this score is provided by a CNN, denoted a single patch network, described below. Then the patch is locally recognizable relative to this score if the inferred class,

$$\hat{c} = \arg\max_c S^c_p$$ (3.1)

is correct.

Note that this notion of recognizability is rather weak. A patch can be small or smooth and get similar scores for several categories. Yet, if the score
associated with the correct category is a bit larger than the other, it is, by this definition, recognizable. The following definition is more meaningful: A patch is \emph{q-locally recognizable} if a categorization procedure accepting this patch as its input classifies it to the correct category with a confidence larger than \( q \). We define these confidence measures in following the sections.

We are also interested in global, image-level, recognizability from an image patch. A sub-image is \emph{globally recognizable} if the correct class score associated with this patch is higher than all other scores associated with all other patches and with other classes. Thus, when the image contains at least one globally recognizable patch, then the estimates class

\[
\hat{c} = \arg\max_c (\max_p S_p^c) \tag{3.2}
\]

is correct.

In contrast to local recognizability, it is unlikely that a very small or smooth patch would be associated with the maximum global score. This is because typically, numerous similar patches will be present in images of other categories. On the other hand, for categories that are not too similar, we expect to find in each image a sub-image of sufficient size and detail that is consistent only with its category.

### 3.2 The Patch-Based Classification Model

The main computational tool developed for the study of minimal recognizable patches is the patch-based classification (PBC) model. The PBC model calculates scores and confidences and performs image-classification based on information included in the best or most informative single patch of the full image. The best patch is unknown and is not pre-specified; therefore, locating it is part of the network’s task. This also means that learning the PBC classifier is a weakly supervised task. The model is aimed at finding the globally minimal patch. Yet, its learning process provides scores and confidences that are used for finding locally minimal patches as well. Aiming at global recognizability is a harder task, and as we found, training for it also produces better classifiers for local recognizability.

We use different networks, sharing a fixed architecture, for each patch size. The PBC model is composed of the three following parts.

(A) Splitting the input image into \( N_p \) (overlapping) spatial patches and resizing each one to a standardized size.
(B) Independently analyzing each patch using a CNN, denoted the single patch network (SPN). For each patch, the SPN provides \( N_c \) patch-level scores, one for each category.

(C) An aggregation layer converting the patch-level scores of all \( N_p \) patches into \( N_c \) image-level scores, which are normalized by a softmax layer, providing \( N_c \) image-level class-probabilities.

See Figure 3.1 for a diagram of the PBC model. We elaborate on each of the different parts comprising this model in the following sections.

### 3.2.1 Single Patch Network

The single patch network takes a single image patch as an input and outputs a vector of \( N_c \) scores, one for each known category (in our tests, \( N_c = 10 \)). The SPN could be any standard classification network. Finding the most accurate network is not a goal of this study. The network we used in our experiments follows the All-Convolutional-Net model [Springenberg et al., 2015]. Relatively simple in nature, this model achieves high accuracy just a little lower than the best classifiers, which are more complex. We modified it slightly by replacing the dropout regularization layers with batch-normalization and the \( 6 \times 6 \) global-averaging layer with a more generalized \( 6 \times 6 \) convolutional layer. The softmax layer was moved out of the SPN, to be placed after the aggregation stage (see Section 3.2.3). A detailed summary of the SPN model architecture is provided in Table 3.1.

Comparing responses and accuracies for different patch sizes is essential in this study. Therefore, to avoid an architecture-dependent bias, we insisted on using a uniform architecture (with different learned weights) and on interpolating the patches to the same input size: \( 32 \times 32 \). As expected, when
experimenting with other interpolated input sizes, we found that smaller interpolated patches work somewhat better with smaller original patch sizes, for which the interpolation is less extreme (see Section A.3). However, the small differences in the accuracy (less than 5%) were not significant for this study.

<table>
<thead>
<tr>
<th>Filter Channel Stride Padding Output ReLU BN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv. Layer 1 3 × 3 96 (1,1) ‘Same’ (32,32,96) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 2 3 × 3 96 (1,1) ‘Same’ (32,32,96) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 3 3 × 3 96 (2,2) ‘Same’ (16,16,96) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 4 3 × 3 192 (1,1) ‘Same’ (16,16,192) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 5 3 × 3 192 (1,1) ‘Same’ (16,16,192) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 6 3 × 3 192 (2,2) ‘Same’ (8,8,192) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 7 3 × 3 192 (1,1) ‘Valid’ (6,6,192) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 8 1 × 1 192 (1,1) ‘Valid’ (6,6,192) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 9 1 × 1 10 (1,1) ‘Valid’ (6,6,10) ✓ ✓</td>
</tr>
<tr>
<td>Conv. Layer 10 6 × 6 10 (1,1) ‘Valid’ (1,1,10) - -</td>
</tr>
</tbody>
</table>

Table 3.1: SPN architecture: “BN” states which layers are followed by a batch-normalization. “Same” padding refers to applying zero-padding on the input to achieve the same spatial size in the output, “Valid” padding means no-padding.

Most experiments in this study were conducted with grayscale images as input. The SPN architecture for grayscale input requires 1,376,652 parameters in memory, out of which 1,374,136 are trainable. The remaining 2,516 parameters belong to the batch normalization layers and are used to memorize running averages of a batch’s mean and standard deviation.

Several preliminary test were performed with color images. A similar SPN architecture is used and only the number of parameters in the first convolutional layer is different.

### 3.2.2 Patch Score Aggregation

The scores for all patches and all categories (patch-level scores) are aggregated to give image-level scores, one for each category. The aggregation method influences the confidence associated with the different categories,
prediction loss, and hence the training process. The impact on the training process is discussed in Section 3.3.1.

For the purpose of finding the most informative patch, we considered two types of max score aggregation:

**Category-independent max** - \( (S_{max-ind}^c) \) This score, evaluated separately for each category, is the maximal score of this category over all patches. For this aggregation, the score for each class is usually taken from a different patch.

**Winner-directed max** - \( (S_{max-dir}^c) \) The image score for all classes is taken from a single patch, the one associated with the overall maximum score.

Both aggregation methods classify the image using the best overall score, as specified in Equation 3.2. The first uses patches that are possibly different from the winner, for evaluating the scores associated with other classes. The second aggregation takes the other classes’ scores from the same patch, ignoring possibly higher scores from other patches. Formally,

\[
S_{max-ind}^c = \max_p \{S_p^c\} \tag{3.3}
\]

\[
S_{max-dir}^c = S_{p^*}^c, \text{ where } p^* = \arg\max_p (\max_c S_p^c) \tag{3.4}
\]

It seems that an intelligent agent wishing to categorize the object(s) in a scene would scan that scene and try to extract the best evidence for each category, no matter where. In this context, calculating confidence using the first category-independent aggregation is justified. On the other hand, when only one patch is observed, the second winner-directed aggregation describes the available information better. Moreover, in scenes containing several objects, the second aggregation allows the detection of multiple categories. For such scenes, it also helps the learning process, because the presence of an object from one category on one place does not indicates that responses to other categories in other locations should be suppressed. Empirically, the two aggregation methods give similar results, with some advantage to the first. See Section 4.2.2 for more details.

Another intuitive aggregation method used in several experiments of this study is the category-independent mean aggregation. Formally, for an image divided into \( N_p \) patches:

\[
S_{mean-ind}^c = \frac{1}{N_p} \sum_{p=1}^{N_p} S_p^c \tag{3.5}
\]
In contrast to the two previous max-aggregations, category-independent mean uses information from the entire image. However, it contains a restriction on the spatial support size of its features, determined by the PBC’s input patch size.

### 3.2.3 Placing the Softmax Layer

The softmax normalization is the final layer, acting on the image-level scores provided by the aggregation layer. In principle, we could alternatively apply softmax normalization on the scores of every patch separately, before aggregation. This however, would let the response to other classes influence the maximum. A substantial but not maximal response to some class, for example, would lower the normalized response to the winning class, which might otherwise be larger than the response to this class in all other patches. See an example to the problematic nature of softmax normalization at each patch in Figure 3.2.

In addition, early experiments with alternative PBC architectures showed that training with softmax normalization to scores at the patch-level hurts the model’s generalization ability, making it more sensitive to overfitting (see Section A.2). This negative effect on generalization was significant with both the category-independent and winner-directed max-aggregations.

**Figure 3.2:** Classification difference between image-level and patch-level scores normalization. Here, there are three possible categories and two image-patches, where \( S_i \) and \( P_i \) are vectors of class scores and probabilities for patch \( i \). With image-level normalization, scores are aggregated and the predicted class is the one associated with the highest score. With patch-level normalization, probabilities are aggregated and the predicted class, in this particular example, is the one with lowest maximal score.

\[
S_1 = \begin{bmatrix} 5 \\ 0.5 \\ 4 \end{bmatrix} \Rightarrow P_1 = softmax(S_1) = \begin{bmatrix} 0.72 \\ 0.01 \\ 0.27 \end{bmatrix}
\]

\[
S_2 = \begin{bmatrix} 0.5 \\ 2.5 \\ 0.5 \end{bmatrix} \Rightarrow P_2 = softmax(S_2) = \begin{bmatrix} 0.11 \\ 0.79 \\ 0.11 \end{bmatrix}
\]
3.3 Implementation

The PBC model was implemented with the Keras and TensorFlow neural-network libraries [Chollet et al., 2015, Abadi et al., 2016].

3.3.1 Training the PBC Model

During training, the spatial stride taken while splitting each image was set to be half the patch size, except for the smallest, $2 \times 2$ patch, for which a $2 \times 2$ stride was used. This was required due to GPU memory limitations and the prolonged training time (see Table 3.2). For evaluation, the stride was always set to be a single pixel.

For training, we used a categorical cross-entropy loss function with an Adam optimizer. A complete training run incorporated 150 epochs, with an initial learning rate of $\eta = 0.001$, reduced by a factor of 2 after every 30 epochs (see Figure 3.3). Weights were regulated by Tikhonov regularization with a $1e^{-4}$ coefficient.

To artificially expand the training dataset and improve performance, data augmentation was utilized via the Keras built-in ImageDataGenerator class. Data augmentation was performed at image-level (prior to splitting into patches) and included random horizontal flips, rotations of $\alpha \in [-10^5, 10^5]$, translations of $\delta \in [-3, 3]$ pixels in both spatial axes, and scaling in the range $\beta \in [0.85, 1.15]$.

The batch size was set to 50 images. However, because each image is split into multiple patches, each resized to the original full-image size of $32 \times 32$ pixels. Practical batch size processed through the model is a function of

![Figure 3.3: Learning rate training schedule.](image)

[Technion - Computer Science Department - M.Sc. Thesis  MSC-2020-34 - 2020]
initial batch size, patch size, and spatial stride. The actual batch sizes for
different patch sizes are provided in Table 3.2.

Training was conducted on either a NVIDIA GeForce Titan X or a NVIDIA
GTX 1080 Ti GPUs, containing 12GB and 11GB of RAM, respectively.
When training with input patches smaller than 10 × 10 pixels, a batch of 50
images surpassed the GPU’s available memory. Thus, in these cases training
was conducted by accumulating gradients of several mini-batches before
weights were updated.

<table>
<thead>
<tr>
<th>Patch Size</th>
<th>Stride Size</th>
<th>Patch Num.</th>
<th>Batch Size</th>
<th>Epoch Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 × 32</td>
<td>(-)</td>
<td>1</td>
<td>50</td>
<td>21.47 sec</td>
</tr>
<tr>
<td>30 × 30</td>
<td>(-)</td>
<td>1</td>
<td>50</td>
<td>24.26 sec</td>
</tr>
<tr>
<td>28 × 28</td>
<td>(-)</td>
<td>1</td>
<td>50</td>
<td>25.03 sec</td>
</tr>
<tr>
<td>26 × 26</td>
<td>(-)</td>
<td>1</td>
<td>50</td>
<td>25.11 sec</td>
</tr>
<tr>
<td>24 × 24</td>
<td>(-)</td>
<td>1</td>
<td>50</td>
<td>25.42 sec</td>
</tr>
<tr>
<td>22 × 22</td>
<td>(-)</td>
<td>1</td>
<td>50</td>
<td>25.56 sec</td>
</tr>
<tr>
<td>20 × 20</td>
<td>(10,10)</td>
<td>4</td>
<td>200</td>
<td>90.86 sec</td>
</tr>
<tr>
<td>18 × 18</td>
<td>(9,9)</td>
<td>4</td>
<td>200</td>
<td>90.66 sec</td>
</tr>
<tr>
<td>16 × 16</td>
<td>(8,8)</td>
<td>9</td>
<td>450</td>
<td>194.74 sec</td>
</tr>
<tr>
<td>14 × 14</td>
<td>(7,7)</td>
<td>9</td>
<td>450</td>
<td>192.64 sec</td>
</tr>
<tr>
<td>12 × 12</td>
<td>(6,6)</td>
<td>16</td>
<td>800</td>
<td>341.75 sec</td>
</tr>
<tr>
<td>10 × 10</td>
<td>(5,5)</td>
<td>25</td>
<td>1,250</td>
<td>510.47 sec</td>
</tr>
<tr>
<td>8 × 8</td>
<td>(4,4)</td>
<td>49</td>
<td>2,450</td>
<td>862.62 sec</td>
</tr>
<tr>
<td>6 × 6</td>
<td>(3,3)</td>
<td>81</td>
<td>4,050</td>
<td>1402.40 sec</td>
</tr>
<tr>
<td>4 × 4</td>
<td>(2,2)</td>
<td>225</td>
<td>11,250</td>
<td>3873.87 sec</td>
</tr>
<tr>
<td>2 × 2</td>
<td>(2,2)</td>
<td>256</td>
<td>12,800</td>
<td>4318.56 sec</td>
</tr>
</tbody>
</table>

Table 3.2: Actual batch size passing through the model during training with
an initial batch size of 50 images.

### 3.3.2 Regularizing the PBC Model

The PBC model has an inherent inner regularization mechanism, due to the
requirement to utilize just a single unknown and not pre-specified patch in
the image for updating the weights, which makes its learning process weakly
supervised.
As a result, we found that models working with smaller patches can benefit from lower regularization. However, the same hyper-parameters were used for the training of all models.

We checked experimentally that optimizing the hyper-parameters depending on the patch size had only a small effect, which is insignificant for this study (see Section A.4).

### 3.3.3 Datasets

There are four datasets used throughout different training schemes and experiments in this study. The first two are the common CIFAR10 and CIFAR100 datasets, originally proposed in [Krizhevsky, 2009].

CIFAR10 contains 60,000 32 × 32 color images, equally distributed between 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. It has a fixed split between a training set of 50,000 samples and a test set of 10,000 (both well-balanced among classes). Each image in CIFAR10 contains a single object, which for most is tightly bounded by the image boundaries.

CIFAR100 also contains 60,000 32 × 32 color images, which can be distributed among either 20 coarse or 100 fine labels. Coarse labels include the following classes: aquatic mammals, fish, flowers, food containers, fruits and vegetables, household electrical devices, household furniture, insects, large carnivores, large man-made outdoor things, large natural outdoor scenes, large omnivores and herbivores, medium-sized mammals, non-insect invertebrates, people, reptiles, small mammals, trees, vehicles 1 and 2.

We started our experiments with the CIFAR10 dataset. Eight of its classes can be divided into pairs of related and similar categories: ship-plane, car-truck, dog-cat and horse-deer. We observed that for small patches, the learned model often preferred one of two similar categories and “gave up” on the second one. It seems that informative small patches of related categories (e.g., the wheels in automobiles and trucks) were effectively indistinguishable for the classifier. By choosing the class with the larger amount or clearer appearance of this patch, the classifier achieves better mean performance. This observation points at a limitation of recognizing from a patch. This phenomenon interferes with finding minimal recognizable patches for the non-preferred categories. Therefore, for the majority of experiments in this study, we experimented with a CIFAR10 variant, which is easier in the sense of containing less inner-similarities. This variant, denoted CIFAR10∗, was based
on classes from the CIFAR10 and CIFAR100 datasets, and consisted of 3,000 samples from each of the following classes: airplane, automobile, bird, cat, deer, frog, fish, tree, person, and insect. The data set was divided into a training set of 25,000 images and a test set of 5,000 images, both well-balanced between the 10 classes.

One final variant of CIFAR10 was constructed for the purpose of evaluating occlusion sensitivity. This dataset, denoted CIFAR10-OCC, contains 9 subsets. Each subset CIFAR10-OCC-\(X\) consists of the CIFAR10 test set, stacked ten times to a total of 100,000 color images and occluded by a randomly placed square mean value patch of size \(32 \times \lfloor X \rfloor\). For example, CIFAR10-OCC-0.4 includes 100,000 images occluded by patches of size \(12 \times 12\).

### 3.3.4 Main Experimental Setup

We used a grayscale version of the CIFAR10∗ dataset to train 16 patch-based models with the first aggregation (category-independent max) for 16 square patch sizes, \(d_i \times d_i\) pixels, where \(d_1 = 32, d_2 = 30, \ldots, d_{15} = 4, d_{16} = 2\). We refer to the models simply as “model of size \(d\)”. The model of size 32 corresponds to the full image. These grayscale input, category-independent aggregation-trained models are the default models used throughout our experiments.

We also trained models with color inputs and using the other aggregations and datasets. These will be noted specifically in the text.
Chapter 4

Patch Size and Recognizability

4.1 Single-Image Recognizability

Ideally, we would like to estimate recognition accuracy as a function of patch size. For a particular image, the accuracy cannot be estimated empirically and is substituted by a single image accuracy estimate, or confidence. In networks trained with cross-entropy loss, the softmax response approximates the posterior probability for this category and may be used as a simple and reasonably accurate confidence [Geifman et al., 2019]. We evaluated correct class confidence as a function of patch size and further measured the confidence difference between two consecutive patch sizes $d_i \rightarrow d_{i+1}$.

The confidence curves for specific images reveal sharp, significant confidence drops in most images. For each image, we refer to the maximal confidence drop associated with two consecutive patch sizes, and denote this shortly as the maximal drop. The curves describing the confidence in the images associated with the largest and the smallest maximal drop (two images for each category) are plotted in Figure 4.1. A small part of the images were associated with a smooth uniform confidence decrease and small maximal confidence drop (see Figure 4.1 (right)). Some other images were difficult to classify even as full images, and were associated with low, smooth confidence curves.

For most images, however, the maximal confidence drop is substantial, as revealed in the maximal drop histogram (see Figure 4.2). Specifically, for the majority of images (3,492 out of 5,000), the maximal drop was larger than 0.5.
Figure 4.1: Confidence curves of images with the largest (left) and smallest (right) maximal drops.

The critical patch sizes associated with the maximal confidence drop are not fixed. Even images of objects from the same category differ a lot due to the intra-class variability and the uncontrolled object pose. To show this variation, we plotted 2D histograms of the maximal drop size and the corresponding (larger) patch size $d_i$ (see Figure 4.3). Clearly, the size of this critical patch varies significantly over the set of images associated with each category, and for some categories there are even several dominant sizes. The images’ maximal drops varied as well, and were typically larger when they occurred with larger patches.

This behavior was reproduced for both grayscale and color images. The average maximal drop was slightly different: 0.608 for color vs. 0.624 for grayscale, and patch size associated with the maximal confidence drop was smaller for color. That is to say, as expected, larger patch sizes are required for recognition from gray level images [Torralba et al., 2008].

The classifier trained with the second winner-directed aggregation led to even more substantial maximal drops: 0.72 on average. The number of images with a maximal drop larger than 0.5 increased as well (3,683 vs. 3,492). This is not surprising, because for each incorrect category, the score associated with the patch corresponding to the winner is, by definition, lower than the score obtained for the best patch in this category. This makes the confidence higher. The confidence drop tends also to be higher because it is a difference between two winner-directed confidences, each of which is higher than the corresponding category-independent confidence.
4.2 Category Recognizability

We carried out additional experiments designed to evaluate the accuracy of categorization from a single patch. This accuracy is estimated simply as the fraction of images for which the PBC model provides the correct label. As expected, smaller patches provide less information and lower accuracy (see Figure 4.4). Remarkably, all categories were classified correctly with 50% accuracy with $12 \times 12$ patches, corresponding to roughly 0.14 of the image.

Considerable variation exists between the curves of different categories as well. Some can be identified from very small patch sizes, which may correspond to either distinct small features (e.g., a wheel) or texture (e.g., tree foliage). Other categories required a coarser scale structure (e.g., birds and cats).
The variability of critical patch sizes within each category, observed in the histograms (Figures 4.2 and 4.3), imply also that different images of the same category may need different patch sizes to be recognized reliably. The fraction of images associated with a sufficiently large patch grows slowly with the patch size, and corresponds to the smooth accuracy curves seen in Figure 4.4. The critical patch size variability (within category) is influenced by the variance of the available instances. Categories with low appearance variance (e.g., cars, which are typically photographed from specific viewpoints) showed higher accuracy over all patch sizes, and in particular, benefit also from small discriminative features.

4.2.1 The Color Information Effect on Categorization

As expected, similar tests with color images, produced higher accuracy and weaker dependency on patch size (see Figure 4.5). Remarkably, all categories were classified correctly with 50% accuracy with $10 \times 10$ patches, corresponding to roughly 0.1 of the image area. Even more remarkable is that the classifier provided 32% mean accuracy with $2 \times 2$ patches.

Color can be very discriminative, especially for the close-set context. For example, in the CIFAR10* dataset, a single blue pixel can hint to one of three classes: fish, birds, or airplanes. See [Torralba et al., 2008] for a study
Figure 4.5: Mean and class-specific categorization accuracy of color images with category-independent aggregation.

revealing the advantage of color in low resolution images.

4.2.2 The Aggregation Effect on Categorization

An important part of the proposed patch-based model is the aggregation, which for the purpose of finding the most informative patch can be either category-independent or winner-directed (see Section 3.2.2). The choice of the aggregation method influences the resulting classification accuracy. Note that the accuracy difference is only due to different training, because once the classifier is trained, the aggregation provides the same winner: the class associated with the highest score in any of the patches (Equation 3.2).

The differences in results between the two aggregations are small (see Figure 4.6). The small difference is still noteworthy, however, because in training with category-independent aggregation, the best patches for each incorrect class are used for suppressing the score to this class. Using these patches and not the particular winner-directed patch is much more informative and leads to better SGD steps, more stable training, and faster convergence. However, even with the less informative winner-directed patches, the obtained accuracy is almost as good. It turns out that with smaller patches, the overlap between these best patches and the winner patch is smaller, the difference is potentially larger, and the disadvantage of learning with winner-dependent
aggregation is more significant. The difference in accuracy is correspondingly larger, but it is still small.

![Mean and class-specific categorization accuracy comparison between classifiers trained with category-independent and winner-directed aggregations.](image)

Figure 4.6: Mean and class-specific categorization accuracy comparison between classifiers trained with category-independent and winner-directed aggregations.
Chapter 5

Minimal Recognizable Image-Patches

5.1 Globally Minimal Recognizable Patches

For all images, the correct classification confidence decreases with the patch size. Let $d^*$ be the minimal patch size for which the image classification is correct. (Such a minimal patch exists for almost all images, with the exception of a few images that are not classified correctly even from a full image). With some abuse of notation, we denote one of these recognizable patches (of size $d^*$) – the one associated with maximal score (and confidence) – as the minimal recognizable patch (MRP).

By definition, the MRP is unique and of globally minimal size. Other globally recognizable image patches of the same critical size $d^*$, but with somewhat lower scores, are often present in the image. As described in Section 4.1, for a majority of images, there is a sharp confidence drop larger than 0.5. Let $(d_i, d_{i+1})$ be the pair of patch sizes associated with this maximal drop. Interestingly, while the MRP is determined exclusively based on recognizability, for most images (3,418 out of 5,000) it is associated with the maximal confidence drop; that is, $d^* = d_i$. Several MRPs are shown in Figure 5.1. Some of them are consistent with human judgment that can identify the category from the MRP but not from the best smaller patch. However, most MRPs are too small for human recognition.
5.2 Locally Minimal Patches

An important motivation for our study was the sharp accuracy drops associated with decreasing patch sizes observed in human vision experiments, but not with recognition algorithms [Ullman et al., 2016] (referred to as the “psychophysical study” in following sections). In our single image recognizability experiments, however, we found sharp confidence drops, but these experiments consider globally minimal patches and are therefore different than the MIRCs considered in both perceptual and computational tests in the psychophysical study.

5.2.1 Computational MIRCs

To get closer to the tests performed in the psychophysical study, we now consider local, MIRC-like patches. Our goal here is to specify these patches, find their distribution in the image, and test whether sharp confidence drops arise between them and their contained patches of smaller size.

We focus on patches associated with correct classification. Thus, we start by using the learned single patch network (SPN) to calculate correct class confidence for every image patch (of every size). It turns out that calculating confidence for every patch is not straightforward. This is because most patches, and especially the smaller ones, are non-informative, while...
the SPN was trained mostly on the most informative patches in the image. Therefore, the SPN gives arbitrary scores for non-informative small patches and the resulting softmax confidence may be occasionally erroneously high, leading to the false conclusion that the patch is informative. Note that even if the classifier would give a low score for all classes, it could be that the softmax ratio would be still high. Thus we take an indirect approach and estimate the confidence as follows: for calculating the correct class confidence of a patch, we take the score (SPN output) from this patch. However, for the incorrect classes, we set the scores as the maximal scores associated with possibly other patches in the image. This is similar to the category-independent (global) aggregation used in the training process. Here, however, as we do not have access to the rest of the image, we substitute these scores with their expected values.

We evaluated the pre-softmax output of the PBC model, associated with the correct and incorrect category classifiers. Interestingly, the maximal scores associated with incorrect classifiers are relatively consistent among different categories and patch sizes (see Figure 5.2 (left)). In particular, small patches get almost constant small scores, as apparent from the low standard deviation. In contrast, correct class scores tend to be more variable.

The observation that false class values are roughly constant, and do not depend on the particular image chosen, allows us to substitute the maximal false scores for each image with their expected values.

We argue that the usage of learned typical scores is legitimate. It makes the decision closer to open set classification and is more consistent with the tests performed in the psychophysical study. It is likely that humans, attempting to recognize an object from a partially visible object, use past experience for calculating their confidence and their final decision. Following the MIRC definition [Ullman et al., 2016], a cMIRC patch is specified as one that is q-locally recognizable, while all its nine contained sub-patches

Figure 5.2: Mean and standard deviation of maximal false scores (left) and maximal correct scores (right) as functions of patch size.
Figure 5.3: cMIRC specification example - a patch that is $q$-locally recognizable (blue), while all its nine contained sub-patches are not (red).

(obtained by cropping each spatial dimension by two pixels) are not. We experimented with several $q \in [0.2, 0.7]$ values and got similar results. The following results correspond to $q = 0.5$.

We found that each image from the CIFAR10* test dataset contains multiple cMIRCs of different sizes and positions. On average, we found 51.5 cMIRCs per image. This number is bigger than the $15.1 \pm 7.6$ MIRCs per image found in the psychophysical study. However, if we exclude cMIRCs with shifts of a single pixel, we revert to 20.6 cMIRCs per image (comparable with the psychophysical study). As expected, we found cMIRCs of several sizes (see Figure 5.4), and one of the smallest cMIRCs typically coincides with the MRP. The average confidence drop between each cMIRC and its best sub-patch (over all images) is 0.64.

Figure 5.4: Mean confidence drop between a cMIRC and its best contained patch (left) and the amount of discovered cMIRCs (right) vs. patch size.

These results are similar to those observed in the psychophysical study, and suggest a simplistic, feedforward model to the perceptual mechanism. An explanation why previous algorithmic attempts did not reproduce the sharp accuracy drop is discussed in the following section.
5.2.2 Evaluating MIRCs with the PBC Model

Typically, MRPs and some of the cMIRCs are relatively small and difficult for humans to recognize (see Figure 5.1). The MIRCs specified in the psychophysical study are more recognizable. We hypothesize that the reason for this difference is the easier tasks considered in our computational study, where close set classification with only 10 categories were considered. In the human study, however, the task was to recognize objects from an unlimited library, which is harder. Therefore, more informative patches, of larger size and detail, were specified as MIRCs. This probably implies that their sub-MIRCs are also recognizable by computational closed set algorithms, which, in turn, imply that the accuracy change, between MIRC and sub-MIRCs, is not significant.

To test this hypothesis, we took two images from [Ullman et al., 2016] that belong to the CIFAR10∗ categories, airplane and bird, and tested all their MIRCs. The MIRC sub-images were estimated manually from the original paper. Each MIRC was resized to a size of $32 \times 32$, and split into 49 sub-MIRC patches (each of size $26 \times 26 - 81\%$ of the original). The correct class confidences and the confidence drop between the MIRC and its best sub-MIRC patch were evaluated using our PBC model. Since MIRCs are originally of different sizes, this was repeated for all PBCs corresponding to all input sizes, and the largest drop was kept. The mean and maximal confidence drop for all MIRCs were 0.11 and 0.3, respectively. These drops are considerably smaller than drops associated with MRPs and cMIRCs, but are consistent with the finding in the psychophysical study.

![Figure 5.5: Two images from [Ullman et al., 2016] with their human-specified MIRCs and computationally specified cMIRCs (excluding highly overlapping patches).](image-url)
Chapter 6
Discussion

This work empirically characterizes the globally minimal sub-image required to categorize an image successfully. A specialized deep network that learns by a weakly supervised auxiliary task was designed for this task. We show that the size this minimal sub-image takes, on average, is a small fraction of its full area, but also that it varies significantly within each category. Following a human vision study [Ullman et al., 2016], another type of minimal recognizable patches that are not globally minimal, but are (locally) minimal in the sense that no sub-patch of them is recognizable, was specified as well.

Both types of minimal recognizable patches share a surprising common property with the human vision study described in [Ullman et al., 2016]: there are image regions that are sufficiently informative for recognition, but which stop providing the required information for recognition following a small size decrease. Moreover, the reduction in region informativeness is sharp and substantial. Remarkably, in both studies, this sharp reduction was not part of the demands but was found, empirically, as a byproduct. Earlier work did not succeed to computationally reproduce the perceptual sharp reduction effect [Ullman et al., 2016].

While the sharp decline in recognition confidence with the patch size (of both types) is similar to the behaviour observed in human vision with MIRC in [Ullman et al., 2016], we do not suggest the simple bottom-up mechanism proposed here as a model for the human visual process. In fact, there is empiric evidence suggesting that humans use top-down processing and even recurrent processing when required to recognize objects from limited information (as in the case of partially occluded objects); see for example [Tang et al., 2018].
Recently, another study used a computational model to specify an additional type of local patches, denoted FRI [Srivastava et al., 2019]. FRI patches are defined by a change in predicted classification output as a result of small shifts in either patch size or position. Therefore, FRI patches are image regions where prediction is sensitive to small changes. However, unlike cMIRC (and MIRC) patches, these image regions are not necessary minimal or recognizable.

In contrast to previous work, which provide specific informative image patches (e.g., the “fragments” in [Vidal-Naquet and Ullman, 2003]) characterizing the images of certain categories, we provide here a generalized and somewhat different characterization: a network that gives a high, discriminative score to the informative patches. Thus, the informative patches themselves are only implicitly characterized, and the best image patches (of the same size and category) may be different in different images.

The minimal recognizable patches (MRPs) we found were small, and usually unrecognizable by humans. This is due, in our opinion, to the closed-set setting and the small number of classes. To estimate MRPs that are more consistent with human vision, more classes or open-set classification tools must be used.
Appendices
Appendix A

Validation of PBC Architecture and Training Process

This appendix includes several additional experiments performed in this study to test the PBC model’s limitations or to validate some of our assumptions and design choices.

In Section A.1, we present the problematic nature of using patch-based classification on a dataset that includes several similar categories. In Section A.2, we demonstrate the accuracy advantage of performing image-level softmax normalization compared to patch-level normalization. In Sections A.3 and A.4, we validate our decisions to use a fixed architecture for all patch sizes and a single set of hyper-parameters.
A.1 Categorizing Images with CIFAR10

For a majority of experiments in this study, we did not use the standard CIFAR10 dataset due to a limitation of recognizing objects from a patch (see Section 3.3.3). Here, we present the categorization accuracy of the PBC model, trained with category-independent aggregation on CIFAR10.

For small patches, the learned model often prefers one of two similar categories and “gives up” on the second one (see Figure A.1). It seems that informative small patches of related categories (e.g., the wheels in automobiles and trucks) were effectively indistinguishable for the classifier. By choosing the class with a larger amount or clearer appearance of this patch, the classifier achieves better mean performance. This observation points at a limitation of recognizing from a patch.

![Figure A.1: Mean and class-specific categorization accuracy of color images from the standard CIFAR10 dataset with category-independent aggregation.](image-url)
A.2 Softmax Placement Experiments

The PBC model architecture described in Section 3.2 consists of aggregating patch-level class scores to image-level class scores, which are normalized by a softmax layer, providing an output of image-level class probabilities. In this section, we refer to this as the “image-level softmax” model.

The alternative option of applying softmax normalization to every patch, prior to choosing the maximal one, would let the response to other classes influence the maximum. We trained six alternative PBC models for six square patch sizes, $d_i \times d_i$ pixels, where $d_1 = 28, d_2 = 24, \ldots, d_5 = 12, d_6 = 8$. These alternative models normalize the class scores of each patch with a softmax operation, before performing category-independent aggregation. We refer to this alternative architecture as the “patch-level softmax” model.

When compared with the standard image-level softmax, patch-level operations reduced accuracy in all tested patch sizes (see Figure A.2). Reduction in accuracy is 0.04 for $28 \times 28$ patches and increase for lower patch sizes up to a maximal difference of 0.17 at $12 \times 12$ patches. Using patch-level softmax operations has a clear negative effect on the training process and the model’s generalization ability.

Figure A.2: Comparing categorization accuracy of image-level softmax and patch-level softmax, category-independent aggregation-trained PBC models.
A.3 Interpolation Effect Experiment

As was stated in Section 3.2.1, to avoid an architecture-dependent bias, we insisted on using a uniform architecture and on interpolating the patches to the same input size: 32 × 32.

To evaluate the effect of this approach, we experimented with other interpolated input sizes: 16 × 16 and 8 × 8. We found that smaller interpolated patches work somewhat better with smaller original patch sizes, for which the interpolation is less extreme (see Table A.1). However, the small differences in the accuracy are not significant for this study.

Note that altering the input size required adjustments to the original SPN architecture (described in Table 3.1). For both 16 × 16 and 8 × 8 interpolation models, the (last) tenth 6 × 6 convolutional layer with linear activation was replaced by a 2 × 2 convolutional layer. In addition, for the 8 × 8 interpolation model, the 3 × 3 kernel of convolution layer no.7 was replaced by a 1 × 1 kernel.

<table>
<thead>
<tr>
<th>Input Patch Size</th>
<th>32</th>
<th>28</th>
<th>24</th>
<th>20</th>
<th>16</th>
<th>12</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interpolation to 32 × 32</td>
<td>0.851</td>
<td>0.849</td>
<td>0.816</td>
<td>0.819</td>
<td>0.794</td>
<td>0.711</td>
<td>0.535</td>
</tr>
<tr>
<td>Interpolation to 16 × 16</td>
<td>0.757</td>
<td>0.786</td>
<td>0.761</td>
<td>0.805</td>
<td>0.812</td>
<td>0.747</td>
<td>0.589</td>
</tr>
<tr>
<td>Interpolation to 8 × 8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.656</td>
<td>0.648</td>
<td>0.575</td>
</tr>
</tbody>
</table>

Table A.1: Categorization accuracy of PBC models with category-independent aggregation for different interpolated input sizes.

A.4 Hyper-Parameters Experiment

The same hyper-parameters are used for training all PBC model (see Section 3.3). To ensure that the effects of optimizing the hyper-parameters for each patch size are minor, we train PBC models with other regularization coefficients and techniques, as well as different initial learning rates. We experimented only with patch sizes \( d_i \in [6, 8, 10, 12] \), which are associated with most maximal accuracy drops.

Categorization accuracy with the CIFAR10* test dataset is summarized in Table A.2. Training with different initial learning rates yields reduced accuracy for all tested patch sizes. Replacing the default ridge regression regularization with lasso regularization also proved to have a negative effect, which was stronger for smaller patch sizes.
As expected, reducing regularization by decreasing the coefficient of ridge regression improved accuracy for smaller patch sizes. The maximal accuracy improvement, associated with a model of $8 \times 8$ patches, is 0.037. This difference is not significant for the purpose of this study.

<table>
<thead>
<tr>
<th>Reg. Tech.</th>
<th>Initial Learn Rate</th>
<th>Reg. Coeff.</th>
<th>Patch Size</th>
<th>12 × 12</th>
<th>10 × 10</th>
<th>8 × 8</th>
<th>6 × 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ridge Regression</td>
<td>0.001 (def.)</td>
<td>1e-3</td>
<td>0.632</td>
<td>0.572</td>
<td>0.441</td>
<td>0.340</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1e-4 (def.)</td>
<td>0.693</td>
<td>0.612</td>
<td>0.503</td>
<td>0.390</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1e-5</td>
<td>0.686</td>
<td><strong>0.632</strong></td>
<td>0.527</td>
<td>0.421</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1e-6</td>
<td>0.699</td>
<td>0.627</td>
<td><strong>0.540</strong></td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td><strong>0.705</strong></td>
<td>0.625</td>
<td>0.529</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.0001</td>
<td>1e-4 (def.)</td>
<td>0.640</td>
<td>0.573</td>
<td>0.504</td>
<td>0.418</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.001 (def.)</td>
<td></td>
<td>0.693</td>
<td>0.612</td>
<td>0.503</td>
<td>0.390</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.01</td>
<td></td>
<td>0.449</td>
<td>0.365</td>
<td>0.307</td>
<td>0.283</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.1</td>
<td></td>
<td>0.100</td>
<td>0.101</td>
<td>0.100</td>
<td>0.100</td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>0.001 (def.)</td>
<td>1e-3</td>
<td>0.371</td>
<td>0.362</td>
<td>0.298</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1e-4 (def.)</td>
<td>0.558</td>
<td>0.488</td>
<td>0.356</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1e-5</td>
<td>0.688</td>
<td>0.622</td>
<td>0.498</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Table A.2: Categorization accuracy of PBC models with category-independent aggregation, trained with different hyper-parameters.
Appendix B

Informative Patch Location Experiment

In our recognizability experiments in Chapters 4 and 5, we use 16 PBC models for 16 square patch sizes $d_i \times d_i$. Each model of size $d_i$ is trained separately.

An optimal model would find the most informative patch in each image and size. Because the difference in patch size between adjacent models is small, the locations of their most recognizable patches is expected to be correlated.

Here, we attempt to characterize the spatial location of the most recognizable patches of different sizes, within each image. The most recognizable patches are specified by the PBC models trained with the category-independent aggregation, similarly to Section 5.1.

As expected, many of the most recognizable patches of size $d_i$ have a considerable overlap with the most recognizable patches of size $d_{i-1}$ (see Figure B.1). This loosely indicates that the most recognizable patch of each size was found by the model.

For patches of size $6 \times 6$ and smaller, spatial consistency between different patch sizes is weaker, which is not surprising since these patches tend to be below MRP size.
Figure B.1: Spatial consistency between most recognizable patches of different sizes in the CIFAR10* dataset. Overlap is calculated relative to the smaller patch area. Average overlap area for each patch size is stated above the x-axis.
Appendix C

Occlusion Robustness Experiments

In this section, we experiment with variants of the PBC model that rely on spatially limited (local) pieces of information for object recognition. We use these model variants to increase the algorithm’s robustness to partial occlusions.

C.1 Benefits of Patch-Based Classification for Occlusion Robustness

A PBC model trained with one of the two max-aggregations is more sensitive to occlusions than standard CNNs, because it learns to classify images based on specific local features. These features may easily be fully occluded and result in a false classification of the image.

Instead, we use the category-independent mean aggregation (see Equation 3.5). Such an approach still limits the spatial support of features used for classification. However, it uses all locally available information in the image, instead of being limited to the single most informative patch.

We used the standard (color) CIFAR10 dataset to train 14 PBC models with the category-independent mean aggregation for 14 square patch sizes, \(d_i \times d_i\) pixels, where \(d_1 = 32, d_2 = 30, \ldots, d_{13} = 8, d_{14} = 6\). The trained models are evaluated with the test sets of CIFAR10 − OCC. Occlusion size is measured by occlusion ratio, which is the relative size of the occlusion’s edge with respect to the image’s edge size (32).
As expected, limiting the spatial support size of the network’s features results in decreased accuracy when tested without occlusions (see Figure C.1). The models associated with the biggest patch sizes, $d_1 = 32, \ldots d_4 = 26$, produce a similar accuracy of approximately 0.93. Correspondingly, the lowest accuracy without occlusions is 0.83, and it is produced by the models working with the two smallest input patch sizes, $d_{13} = 8, d_{14} = 6$. This decrease in recognition is 9% relative to model $d_1 = 32$. Note that model $d_1 = 32$ is a standard CNN model, working with full-images and not patches.

Unsurprisingly, recognition accuracy decreases for all models as occlusion ratios increase. However, the decrease is not uniform for all input patch sizes. With an occlusion ratio of 0.5, the accuracy of model $d_1 = 32$ is 0.49, while the accuracy of model $d_{13} = 8$ stands at 0.65, higher by 40%. Furthermore, the best results (on average) are achieved when using this PBC model with $8 \times 8$ input patches, which provides top accuracy in three of the nine tested occlusion sizes and is second best for two more.

We repeat the experiment for classifiers trained with the CIFAR100 dataset and observe similar results, characterized with an even larger relative advantage for the PBC models using small patch sizes. When comparing to the
full image model $d_1 = 32$, worst accuracy decrease without occlusions is 25% while the biggest improvement under partial occlusions is close to 75% (see Figure C.2).

[Figure C.2: Categorization accuracy of PBC models, trained on CIFAR100 with category-independent mean aggregation, as a function of the occlusion ratio.]

C.2 Regularization for Occlusion Robustness

Our experiments show that limiting the spatial support size of the network’s features results in improved robustness to occlusions. These results are similar to findings in a recent related study [Osherov and Lindenbaum, 2017].

To compare our method with the one proposed in [Osherov and Lindenbaum, 2017], we implement the central-moment (CM) regularization, which produced the best results in the original study, and use it to train 3 PBC models with the category-independent mean aggregation for patch sizes, $d_1 = 32, d_2 = 16, d_3 = 8$. We performed a grid search of the optimal CM regularization coefficient for each input patch size and report the results of the best models. We compare these results with the results associated with the PBC models described in the previous section (trained with standard L2 regularization).
We examined two additional regularization techniques that can be used to reduce the spatial support of a model’s learned features. The random-erasing (RE) regularization, proposed in [Zhong et al., 2017], selects random rectangular areas in images and replaces them with random values during training. Note that this can be considered as training with occlusions, which is less desired because it has been shown to not generalize well to different types of occlusions (see [Pepik et al., 2015]). The second regularization technique is proposed by us and called patch-swap (PS) regularization. During training, the algorithm selects random pairs of square areas in the image and swaps between their pixel values. The original data is not occluded but shuffled, potentially encouraging the model to learn features with smaller spatial support. Both RE and PS techniques require the tuning of multiple hyper-parameters and we performed an extensive grid search and report the results of the best models.

We evaluate the full-image PBC models (i.e., \( d = 32 \)) trained with the different regularization techniques with the test sets of CIFAR10 − OCC. All three techniques help improve the robustness to occlusions of a standard CNN model (see Figure C.3). The accuracies of models trained with, either the PS or RE regularization techniques, are almost indistinguishable. The overall least sensitive model to occlusions of different sizes is the one trained with CM regularization. However, for occlusions of intermediate sizes (i.e., occlusion ratios \([0.4, 0.5, 0.6]\)), a standard L2 regulated PBC model with \(8 \times 8\) input patches produces the highest accuracy (see Figure C.3).

Unsurprisingly, applying CM regularization combined with patch-based classification fails to effectively improve on the results of a CM regulated full-image CNN. See results of PBC \(8 \times 8\) and \(16 \times 16\), trained with CM regularization in Figure C.3.

C.3 Multiple Patch-Size PBC Models

In our experiments we can see that the highest accuracy for different occlusion ratios is associated with different models. If we could estimate an input image’s occlusion ratio, we could choose the better fitting model and benefit from optimal accuracy for all occlusion sizes.

For this purpose, we examine the score output (prior to softmax normalization) of category-independent mean aggregation PBC models associated with input patch sizes \(d_1 = 32, d_2 = 16, d_3 = 8\), for different occlusion ratios.
Figure C.3: Categorization accuracy of PBC-mean models, trained on CIFAR10 with different regularization techniques, as a function of the occlusion ratio.

Following this examination, we note that:

- Occlusions reduce the scores output for all class, both false and true.
- The score decrease for the true class is (on average) the largest.
- Score’s variance increases with smaller patch sizes.

See Figure C.5 for an example of the score histograms of images from the ”car” class in the test sets of CIFAR10 – OCC.

Since the scores associated with the full-image PBC model have the lowest variance, we use them to set thresholds that determines whether an input image contains an occlusion. For each new image, we process it with the full-image PBC model and compare its maximal score with its class-specific threshold value. If the score is bigger than the threshold, the image is categorized to the corresponding class, if not, than we use the PBC $8 \times 8$ model to classify it.

We set for each class a threshold value that corresponds to 0.42 of its unoccluded training set TP scores mean. This value was set based on the highest overall accuracy with the CIFAR10 – OCC training set.
We evaluate the resulting algorithm on the CIFAR10 – OCC test set. Applying this simple thresholding algorithm achieves improved robustness to occlusions without any decrease in accuracy for unoccluded images. This algorithm outperforms all models and regularization techniques that we evaluated in this study (see Figure C.4).

Figure C.4: Categorization accuracy of the multiple patch-size input PBC model as a function of the occlusion ratio.
Figure C.5: Example of the scores histograms for images of class 1 in the test sets of CIFAR10—$OCC$. The scores associated with the full-image model are red and those associated with the PCB-4 $\times$ 8 model are blue. Each column corresponds to the model’s output for a specific class and each row to the occlusion ratio of the tested images.
Bibliography


打ち上げ וחולשת במירבולי הניווטן לדיארה

מרק פונרובה
 поиск הטלאי המינימלי הניתן לזיהוי

חבר על מחקר
לשםميلו שלחקים לעהרישת קיבולות החזון
מגייסים לפיזיוס במערכתอารมוניות וה,List
מקס פונדרו

הוגש לשם טכניקות – מכנון טכנולוגיות לינאטא
סיון משה"מ, חיפה, יוני 2020
המחקר נעשה בהנחיית פרופסור Michaël Lindenbaum, במגזרה تقومית הבינית ע"י
יחידה למש.addAll אוטונומיות ורובוטיקה בטכניון.
אני מודה לקרן ג'ייקובס וטכניון על התמיכה הכספית הנדרשת בהשתלשלותי.
Algorithms based on artificial neural networks have been responsible for several breakthroughs in recent years and are currently at the forefront of technology in computer vision, with a strong emphasis on object recognition. However, despite tremendous progress and continually improving performance, these algorithms still fail to match human ability in recognizing small or partially hidden objects. Performance of artificial neural networks is compromised when a partially hidden or limited view of the object is presented.

Recently, a psychophysical experiment involving thousands of participants was conducted by Ulman and co-workers. This experiment involved minimal recognizable configurations (MIRC), defined as the smallest configuration that allows for recognition. The experiment began with ten complete images from different categories. Each correctly recognized image by more than 50% of the participants was divided into five new images. Four of these images were smaller by 20%, and the fifth image was of lower resolution by 20%. Each image was shown to new participants until less than half of the participants could correctly identify the object in the image. Images that were successfully identified but none of the five images included them were identified as MIRC.

One of the surprising findings in the work of Ulman and co-workers is that human recognition performance is significantly reduced when moving from MIRC to smaller configurations. This phenomenon was not observed in similar tests conducted using different computer vision algorithms, where accuracy showed a gradual and consistent decline. This indicates that human recognition involves a different mechanism compared to artificial neural networks.

Following these results, we are interested in the natural question: what is the smallest area of the image that allows for the identification of the object?

In this work, we address a slightly more practical question: what is the minimum square area that allows for object identification when using a neural network? We use this algorithm as a replacement for general recognition, as at this moment, artificial neural networks provide the best results in certain tasks, even exceeding human capability.

We investigate this question for two types of MIRC:
- Global minimal recognizable patch (MRP): We are looking for the smallest patch that allows for the correct classification of the image. We call this patch minimal recognizable patch (MRP).
- Local minimal recognizable patch (LRP): We are looking for a patch that allows for the correct classification of the image, but any smaller patch included in it does not allow for classification. This criterion corresponds to the definition of MIRC, but in our work, we define patches using a computer algorithm and refer to them as computational MIRC (cMIRC).

To determine the location and characteristics of these patches, we developed a unique architecture of neural networks that locate the most informative area and classify the image based on the information contained in this area alone. We tested several versions of this architecture, named PBC, which vary in patch size and methods of local information collection. The minimal patches we found vary between and within different categories, and their size increases for higher accuracy requirements.

We find these patches through use of a grayscale image database of size 32 × 32 pixels, named CIFAR. Each image in this database contains an object.
A unit which is almost exactly enclosed by a border is a candidate for an object. The central tool we use to find these objects is the PBC:

- A three-level network is constructed. The first level is encoded into a grid of 32 x 32 pixels. Each level is passed through a level-specific network, generating a vector of classification indicators for each object. Then, the classification vectors of each object are passed to the aggregation layer, which converts them to a classification vector for the image. The vector is then normalized using the softmax function to obtain a vector of probabilities for each category at the image level.

After testing several networks with different object sizes from 2 x 2 to 6 x 6 pixels, we found that the most common change in performance was visible for objects of a certain size. This size is not constant and varies between images and within certain categories. Therefore, when we tested similar accuracy for classification as a function of object size, we did not observe any significant changes.

This experimental method helped us find the minimal objects. MRP is the smallest object that can localize the correct classification. cMIRC is the object that is identified with high confidence greater than 0.5, but none of the sub-objects identified with this confidence for the correct category. The two minimal objects we defined are characterized by sharp declines above 0.6. This result is similar to the average of the expected 0.57% accuracy, as determined in the psychological experiment.

The minimal objects we found were usually too small for human vision. Since we only have 10 categories in our image bank, the classification task is relatively simple and can be performed with small objects. However, to increase the accuracy of the system and increase the challenge, we are required to use different categories or open-set classification methods.

In summary, our contribution to this study is threefold:

1. We propose a unique neural network architecture that uses individual objects to classify images. In addition, we define two types of minimal objects that allow us to classify images.
2. We conduct a basic study of graph behavior as a function of object size. This study points out several similarities between the human vision system and our system's implementation, when using objects for classification.

This report, summarizing the author's research into the thesis, is structured as follows:

Chapter 1 introduces the general topic of the study. In Chapter 2, we provide the necessary mathematical background and a review of relevant works related to the minimal information required for recognition, both for human and computer algorithms.

Chapter 3 describes the main tools we use in this research, with a focus on the artificial neural network we defined for this research, including a detailed overview of each part and the learning process. Chapters 4 and 5 describe the main experiments conducted in this research. First, experiments characterizing the network's ability to classify objects as a function of object size and then experiments to find the minimal objects.

Chapter 6 summarizes the main conclusions drawn from this study. Additional experiment results that are not relevant to understanding the study's conclusions are presented in the appendices.