Universidade Estadual de Campinas
Instituto de Computação

Alan Zanoni Peixinho

Learning Image Features by Convolutional Networks under Supervised Data Constraint

Aprendendo Características de Imagem por Redes Convolucionais sob Restrição de Dados Supervisionados

CAMPINAS
2017

Alan Zanoni Peixinho

Learning Image Features by Convolutional Networks under Supervised Data Constraint

Aprendendo Características de Imagem por Redes Convolucionais sob Restrição de Dados Supervisionados

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Thesis presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Computer Science.

Supervisor/Orientador: Prof. Dr. Alexandre Xavier Falcão

Este exemplar corresponde à versão final da Dissertação defendida por Alan Zanoni Peixinho e orientada pelo Prof. Dr. Alexandre Xavier Falcão.

CAMPINAS

2017

Agência(s) de fomento e nº(s) de processo(s): CAPES; CNPq

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

P359L

Peixinho, Alan Zanoni, 1990-
Learning image features by convolutional networks under supervised data constraint / Alan Zanoni Peixinho. – Campinas, SP : [s.n.], 2017.

Orientador: Alexandre Xavier Falcão.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Redes neurais (Computação). 2. Análise de imagem. 3. Aprendizado de máquina. I. Falcão, Alexandre Xavier,1966-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Aprendendo características de imagens por redes convolucionais sob restrição de dados supervisionados
Palavras-chave em inglês:
Neural networks (Computer science)
Image analysis
Machine learning
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Alexandre Xavier Falcão [Orientador]
Hélio Pedrini
João Paulo Papa
Data de defesa: 05-05-2017
Programa de Pós-Graduação: Ciência da Computação

Universidade Estadual de Campinas
Instituto de Computação

Alan Zanoni Peixinho

Learning Image Features by Convolutional Networks under Supervised Data Constraint

Aprendendo Características de Imagem por Redes Convolucionais sob Restrição de Dados Supervisionados

Banca Examinadora:

- Prof. Dr. Alexandre Xavier Falcão (Presidente)
  Instituto de Computação - UNICAMP
- Prof. Dr. Hélio Pedrini
  Instituto de Computação - UNICAMP
- Prof. Dr. João Paulo Papa
  Faculdade de Ciências - UNESP

A ata da defesa com as respectivas assinaturas dos membros da banca encontra-se no processo de vida acadêmica do aluno.

Campinas, 05 de maio de 2017

"I love deadlines. I love the whooshing noise they make as they go by."

—Douglas Adams

Agradecimentos

Primeiramente, me sinto na imprescindível obrigação de agradecer aos meus pais, os quais, mais do que nunca, mostraram seu suporte incondicional nesses momentos difíceis. Espero que, um dia, eu seja capaz de retornar, mesmo que minimamente, todo o afeto, carinho, sabedoria e orientação que me deram.

Ao professor Falcão, um pesquisador de talento inquestionável, que sempre se preocupou de forma genuína com o desenvolvimento de todos os seus alunos, e se desdobrou de forma ímpar, para estar presente de forma direta no desenvolvimento deste trabalho. Agradeço muito por todo o direcionamento acadêmico, filosófico e pessoal. E, principalmente, por enxergar potencial em mim, onde nem mesmo eu o posso.

Aos companheiros de laboratório, tanto do LIDS, quanto do LIV, agradeço pelos muitos cafés, discussões, e aprendizado compartilhado que proporcionaram. Com um agradecimento especial ao Samuel e Deângeli, que me ajudaram, e muito, na revisão deste texto.

Tive ainda o prazer de fazer bons amigos, com uma incrível diversidade de ideias e comportamentos, os quais gostaria de agradecer. Saibam que, cada um a sua maneira, vocês deixaram um pedaço de suas personalidades comigo.

Por fim, deixo minha gratidão à Unicamp, que acolhe aos seus alunos com uma estrutura de excelência. Bem como ao fomento à pesquisa fornecidos pela CAPES/CNPq, que tornaram este trabalho possível.

Abstract

Image analysis has been widely employed in many areas of the Sciences and Engineering to extract and interpret high-level information from images, with applications ranging from a simple bar code analysis to the diagnosis of diseases. However, the state-of-the-art solutions based on deep learning often require a training set with a high number of annotated (labeled) examples. This may imply significant human effort in sample identification, isolation, and labeling from large image databases, specially when image annotation asks for specialists in the application domain, such as in Medicine and Agriculture, such requirement constitutes a crucial drawback. In this context, Convolution Networks (ConvNets) are among the most successful approaches for image feature extraction, such that their combination with a Multi-Layer Perceptron (MLP) network or a Support Vector Machine (SVM) can be used for effective sample classification. Another problem in these techniques is the resulting high-dimension feature space, which makes difficult the analysis of the sample distribution by the commonly used distance-based data clustering and visualization methods. In this work, we analyze both problems by assessing the main strategies for ConvNet design, namely Architecture Learning (AL), Filter Learning (FL), and Transfer Learning (TL), according to their capability of learning from a limited number of labeled examples, and by evaluating the impact of feature space reduction techniques in distance-based data classification and visualization. In order to confirm the effectiveness of feature learning, we analyze the progress of the classifier as the number of supervised samples increase during active learning. Data augmentation has also been evaluated as a potential strategy to cope with the absence of labeled examples. Finally, we demonstrate the main results of the work for a real application — the diagnosis of intestinal parasites — in comparison to the state-of-the-art image descriptors. In conclusion, TL has shown to be the best strategy, under supervised data constraint, whenever we count with a learned network that suits the problem. When this is not the case, AL comes as the second best alternative. We have also observed the effectiveness of Linear Discriminant Analysis (LDA) in considerably reducing the feature space created by ConvNets to allow a better understanding of the feature learning and active learning processes by the expert through data visualization. This important result suggests an interplaying between feature and active learning with intervening of the experts to improve both processes as future work.

Resumo

A análise de imagens vem sendo largamente aplicada em diversas áreas das Ciências e Engenharia, com o intuito de extrair e interpretar o conteúdo de interesse em aplicações que variam de uma simple análise de códigos de barras ao diagnóstico automatizado de doenças. Entretanto, as soluções do Estado da Arte baseadas em redes neurais com múltiplas camadas usualmente requerem um elevado número de amostras anotadas (rotuladas), implicando em um considerável esforço humano na identificação, isolamento, e anotação dessas amostras em grandes bases de dados. O problema é agravado quando tal anotação requer especialistas no domínio da aplicação, tal como em Medicina e Agricultura, constituindo um inconveniente crucial em tais aplicações. Neste contexto, as Redes de Convolução (*Convolution Networks* - ConvNets), estão entre as abordagens mais bem sucedidas na extração de características de imagens, tal que, sua associação com Perceptrons Multi-Camadas (*Multi Layer Perceptron* - MLP) ou Máquinas de Vetores de Suporte (*Support Vector Machines* - SVM) permite uma classificação de amostras bastante efetiva. Outro problema importante de tais técnicas se encontra na alta dimensionalidade de suas características, que dificulta o processo de análise da distribuição das amostras por métodos baseados em distância Euclidiana, como agrupamento e visualização de dados multidimensionais. Considerando tais problemas, avaliamos as principais estratégias no projeto de ConvNets, a saber, Aprendizado de Arquitetura (*Architecture Learning* - AL), Aprendizado de Filtros (*Filter Learning* - FL) e Aprendizado por Transferência de Domínio (*Transfer Learning* - TL) em relação a sua capacidade de aprendizado num conjunto limitado de amostras anotadas. E, para confirmar a eficácia no aprendizado de características, analisamos a melhoria do classificador conforme o número de amostras aumenta durante o aprendizado ativo. Métodos de *data augmentation* também foram avaliados como uma potencial estratégia para lidar com a ausência de amostras anotadas. Finalmente, apresentamos os principais resultados do trabalho numa aplicação real — o diagnóstico de parasitos intestinais — em comparação com os descritores do Estado da Arte. Por fim, pudemos concluir que TL se apresenta como a melhor estratégia, sob restrição de dados supervisionados, sempre que tivermos uma rede previamente aprendida que se aplique ao problema em questão. Caso contrário, AL se apresenta como a segunda melhor alternativa. Pudemos ainda observar a eficácia da Análise Discriminante Linear (*Linear Discriminant Analysis* - LDA) em reduzir consideravelmente o espaço de características criado pelas ConvNets, permitindo uma melhor compreensão dos especialistas sobre os processos de aprendizado de características e aprendizado ativo, por meio de técnicas de visualização de dados multidimensionais. Estes importantes resultados sugerem que uma interação entre aprendizado de características, aprendizado ativo, e especialistas, pode beneficiar consideravelmente o aprendizado de máquina.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

"Understand well as I may, my comprehension can only be an infinitesimal fraction of all I want to understand."

—Ada Lovelace

In many areas of Sciences and Engineering, images can provide important information about real problems, such as diagnosis of diseases, biometric access to a service by image analysis, and quality control of industrial products based on image inspection. This image analysis usually requires an initial step of manual *isolation* (identification and/or segmentation) and *identification* (label assignment) of the content of interest, named *sample*, in training images. Such samples may appear as (Figure 1.1):

i. *pixels*, the constituent image elements;

ii. *superpixels*, connected regions that transmit a same color and texture visual sensation;

iii. *objects*, connected regions with known shape, or

iv. *subimages* around regions of interest.

The advance of imaging and storage devices favor the acquisition of unlabeled images in large scale. In some applications, such as the diagnosis of human intestinal parasites (enteroparasites) from optical microscopy image analysis [88], each image may also contain a high number of samples — objects to be identified as either fecal impurity (the great majority) or one among the 15 most common species of human enteroparasites in Brazil. A single microscopy slide with 22×22 mm^2 may contain, for instance, 2.2 thousand images containing about 100 thousand of such samples each. In such cases, manual labeling by experts of a high number of samples for supervised machine learning can be infeasible, especially when crowd-sourcing techniques [16] cannot be applied.

For a given training set with samples labeled by experts, the supervised learning process relies on mathematical characterization (*image features*) of each sample to design

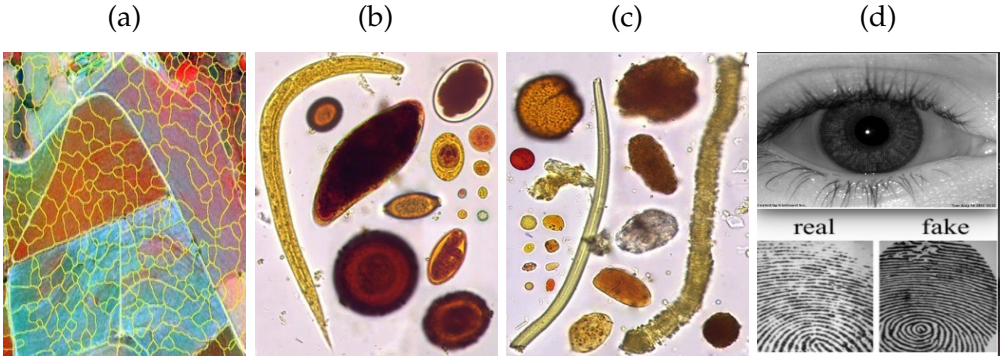


Figure 1.1: (a) Superpixels of a remote sensing image identified as either coffee or non-coffee plantation region. (b) Objects from optical microscopy — fecal impurities and species of human enteroparasites — for classification, being (c) several impurities similar to the parasites. (d) Subimages around iris and fingerprints for person identification.

a *pattern classifier* — a function that assigns labels to new samples. Therefore, the supervised learning process encompasses both tasks i.e., the feature learning process and the design of the pattern classifier.

1.1 Scope of the work

The present work assumes sample isolation can be automated and concentrates on the supervised learning process. For the sake of feasibility, this work also assumes the number of training samples labeled (supervised) by experts is limited to a reasonably low value with respect to the number of available unlabeled samples. Each training sample s can be characterized by a set of $n \geqslant 1$ measures extracted from the image, which maps s into a vector $\vec{x}(s) \in \mathbb{R}^n$ (or point) of the corresponding n-dimensional feature space. The process of learning a good set of measures is called *feature learning*, which selects a feature space where samples from different labels (*categories*) fall in separable regions by a pattern classifier.

Methods for feature learning can be divided into *knowledge-based* and *data-driven* approaches. Knowledge-based methods count on experts knowledge about the problem to develop a suitable feature extraction algorithm [4]. In the diagnosis of Alzheimer disease from brain MR-image analysis, for instance, features of interest are expected to capture some abnormal hippocampal atrophy [37]. Such handcrafted features are usually more meaningful to experts of the application domain and therefore less susceptible to generate a pattern classifier that overfits the training data [1] than features from the data-driven approaches. On the other hand, data-driven methods, such as those that rely on neuron network architectures with multiple hidden layers —

[1] A classifier with considerably reduced power to predict the labels of new samples.

a strategy well known as *deep learning* —, have been shown to be more effective than knowledge-based approaches [46, 59, 68]. Data-driven methods rely on some optimization process to exploit correlation between features extracted from the samples and the corresponding sample labels. Such approaches can be further divided into *filter* and *wrapper* methods [33], where filters select features independently of the pattern classifier, while wrappers design the feature extraction algorithm and the classifier at the same time, by using the performance of the classifier on part of the training set to score the predictive power of the features.

Many deep learning techniques can be seen as a wrapper that embraces a wide family of different techniques based on the concept of stacking several neural network layers. Among these techniques, Convolution Networks (ConvNets) present the best results to extract features from subimages around detected/segmented objects for image classification in many different scenarios [19, 32, 46, 52]. These ConvNets consist of a sequence of layers, with linear and non-linear operations each, to transform low level image features (e.g., pixel intensities) into high level texture features, more suitable for pattern classification. As we will discuss in this work, the output of the last convolution layer can be interpreted as a *high-dimensional feature vector*. Thus following it, one may consider either a decision layer (classifier) or more network layers for feature space dimensionality reduction followed by a decision layer — i.e., the traditional multi-layer perceptron network [36].

However, feature learning through ConvNets is known to require a high number of supervised samples. Therefore, the scope of this work is reduced to study **image feature learning by ConvNets under supervised data constraint**. Under such constraint, for the cases where the pattern classifier is able to achieve high accuracy rates in a test set, we can certainly conclude the descriptor is less prone to overfit.

1.2 Image feature learning by ConvNets

The name ConvNet stems from the operation performed by the first layers of such networks, which relies on a linear convolution between input data and a filter bank — a considerable simplification of the first operation in a typical neural network layer, since the general case defines a different weight set for each neuron. In this operation, each pixel can be interpreted as a neuron with the same synaptic weights for all pixels of the same layer. The filter coefficients are those synaptic weights and the filter size, which is the same for all filters in the layer, defines the neuron receptive field. The adjacent pixel values form the input data of each neuron. A ConvNet layer also consists of additional operations, namely *activation*, *pooling*, and *normalization*. Therefore, learn the best features for a problem consists of finding the best solution for the number of layers, number of filters per layer, their coefficients, and the parameters of each remaining operation. These are called the network *hyperparameters*, with exception of the filter coefficients which are called the network *parameters*.

The general pipeline for feature learning based on ConvNets is shown in Figure 1.2.

The input data is a subimage (supervised sample) and the output data is a feature vector in some n-dimensional feature space, which is usually high and sparse. For the sake of feasibility, the existing feature learning techniques optimize either the choice of the network parameters or the choice of its hyperparameters, and the classification errors in a training/evaluation set are used as optimization criterion.

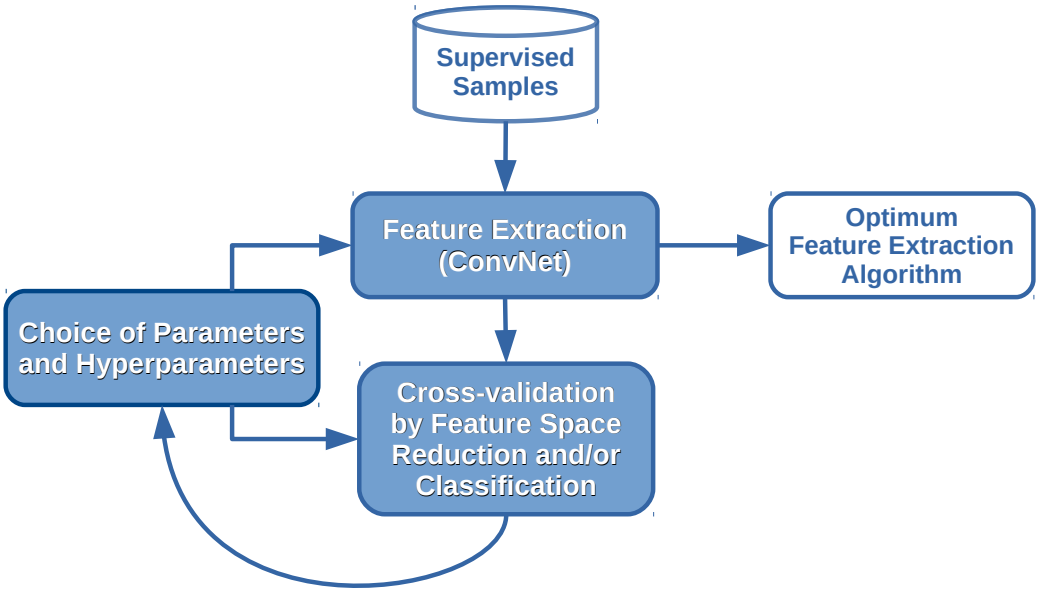


Figure 1.2: General pipeline for image feature learning based on ConvNets.

As previously mentioned, one can use directly the output features of the ConvNet as input to a decision layer — e.g., a classifier based on logistic regression or support vector machines [36]. Optionally the output features of the ConvNet may be used as input to a multi-layer perceptron, when aiming to reduce dimensionality and specialize neurons for each category [74] before the decision layer. Thus, the weights for convolution, dimensionality reduction, and decision layers are called the network *parameters*, in contrast to the *hyperparameters*, previously mentioned.

Feature learning strategies based on ConvNets may be divided as follows. The most common one fixes the network architecture and learn the weights of all network layers [10]. We call this strategy *Filter Learning*. In Filter Learning, it is also possible to set up the weights of the network with values learned from a different image classification problem, as a starting point, and refine them with a back-propagation gradient descent algorithm. This strategy is called *Transfer Learning* [64, 95], also known as *fine tunning*. And, finally, one can learn the best network architecture (*Architecture Learning*), by setting the weights of the ConvNet randomly [69].

1.3 Objectives and methodology

In order to study feature learning based on ConvNets under supervised data constraint, we must consider the popular existing strategies, select image data sets with the various distinct characteristics (e.g., type of sample, balanced and unbalanced number of samples per category, image domain, and number of categories), and establish a reasonably low number of supervised samples for the feature learning process. Under these conditions, we wish to identify the best feature learning strategy with an empirical analysis.

On the proposed scenario of supervised data constraints, we want to figure out which is the best approach for feature learning, among *Filter Learning*, *Transfer Learning* and *Architecture Learning*, by imposing a strong limitation on the ammount of supervised data available for ConvNet feature learning.

In order to confirm whether or not this feature learning process was effective, we must also decouple the feature extraction algorithm from the pattern classifier, and evaluate the performance of the latter, with the former being fixed, as the number of supervised samples increases. We also evaluates, specifically for Architecture Learning, how increasing the number of samples impacts the descriptor and classifier, separately. Considering the output of ConvNets usually consists of a high-dimensional feature vector, harming some common operations (e.g., data clustering, distance-based classification, content-based image retrieval) that depend on computing distance functions among samples, therefore we want to evaluate the effectiveness of the dimensionality space reduction strategies as well.

Among the available strategies to deal with supervised data constraint, we can include the possibility of augmenting the training set with unsupervised samples (also known as *data augmentation*). For this reason, we also want to asses the effect of augmenting the number of samples in the performance of ConvNets, considering artificial data augmentation strategies, as well as the inclusion of the expert in the learning process.

Finally, we wish to validate the best solution in a real problem and, for that purpose, we have selected data sets from the diagnosis of intestinal parasites. Such data sets can be automatically obtained when processing images from the microscopy slides [88].

These questions are seen in more details on Chapter 3.

1.4 Main contributions

Preliminary results for the diagnosis of intestinal parasites were published in [66]. As it will be shown, we have found the best strategy to handle supervised data constraint among the studied ones, while evaluating important limitations and characteristics of ConvNets on such restrictions. After all, we validated our results on the diagnosis of intestinal parasites in comparison with the state-of-the-art handcrafted image features from [88] and our own ConvNet-based approach from [66].

The fact we decoupled the feature extraction algorithm and the pattern classifier to evaluate the effectiveness of the feature learning process, revealed the importance of exploiting in future work an interplay between feature learning and *active learning* [53] — a process that uses an apprentice classifier to label and discover the most relevant samples **in a given feature space** for expert supervision along learning iterations, in which the classifier is retrained with a higher subsequent number of supervised samples. In the beginning of this process, the samples that better represent the categories are necessary to teach the classifier the rough location of the decision boundaries between categories. In the remaining learning iterations, the samples near those decision boundaries (the most informative ones) are needed to adjust them finely, by minimizing classification error in an evaluation set. Current solutions, named active feature learning [97, 98], do not decouple feature extraction and pattern classification, and so the samples selected in a given feature space might not be the most relevant ones to retrain the neural network. By decoupling feature extraction and pattern classification, we believe the active learning process will reveal when the selected feature space is suitable for the problem. At the same time, a question remains unsolved: how can we identify an initial set of relevant samples for feature learning? Currently, the expert is responsible for that. If they are suitable, they should be enough to select a feature space in which, by using multidimensional data projection [74], the expert can identify more representative samples for feature learning.

1.5 Organization of the text

Chapter 2 introduces the main concepts, terminology, and provides an overview about popular Deep Learning methods. Chapter 3 presents the proposed methodology to evaluate feature learning by ConvNets under supervised data constraint. The considered ConvNet-based strategies for feature learning are evaluated in Chapter 4, where the best approach in feature learning is validated in a real application — the diagnosis of intestinal parasites — in Chapter 5. Chapter 6 states the conclusion by discussing the main results and indicating the most promising strategy to be investigated in future work.

Chapter 2

Background

"Deep learning is just a buzzword for neural nets, and neural nets are just a stack of matrix-vector multiplications, interleaved with some non-linearities. No magic there."

—Ronan Collobert

This chapter provides the main concepts and terminology to understand the related methods and remaining chapters.

2.1 Problem definition

For any given set $\mathcal{Z}$ of samples from c distinct categories of a pattern recognition problem, we wish to discover a function (model) $M\colon \mathcal{Z} \to \{1, 2, \ldots, c\}$ that can assign to any sample $s \in \mathcal{Z}$ the label $\Omega(s) \in \{1, 2, \ldots, c\}$ of its corresponding category [13]. In supervised pattern recognition, M is estimated based on a set $\mathcal{X}(s) = \{x_1(s), x_2(s), \ldots, x_n(s)\}$ of n observations (measures or features) about each sample s of a training set $\mathcal{Z}_1 \subset \mathcal{Z}$, for which the label $\Omega(s)$ is known *a priori*. One can also represent the feature set $\mathcal{X}(s)$ as a vector $\vec{x}(s) = (x_1(s), x_2(s), \ldots, x_n(s)) \in \mathbb{R}^n$ (or point) of the corresponding n-dimensional feature space. Hence, the solution of the problem essentially consists of finding a **feature extraction algorithm** $\mathcal{X}$ that maps samples $s \in \mathcal{Z}$ into separable regions of the feature space $\mathbb{R}^n$. The model M must then be essentially designed to minimize the classification errors (i.e., when the label $L(s)$ assigned by M is different from $\Omega(s)$), which might occur whenever the success of $\mathcal{X}$ is incomplete and/or the design of M is imperfect. That is, the model M is expected to maximize the posterior probability $P(L(s) = \Omega(s) \backslash \vec{x}(s))$ of the correct decision over the samples $s \in \mathcal{Z}_1$. Feature learning is then concerned with the design of the feature extraction algorithm, but deep learning approaches put together in the same pipeline the design of $\mathcal{X}$ and M. The general (traditional) pattern recognition pipeline, however, is illustrated in Figure 2.1.

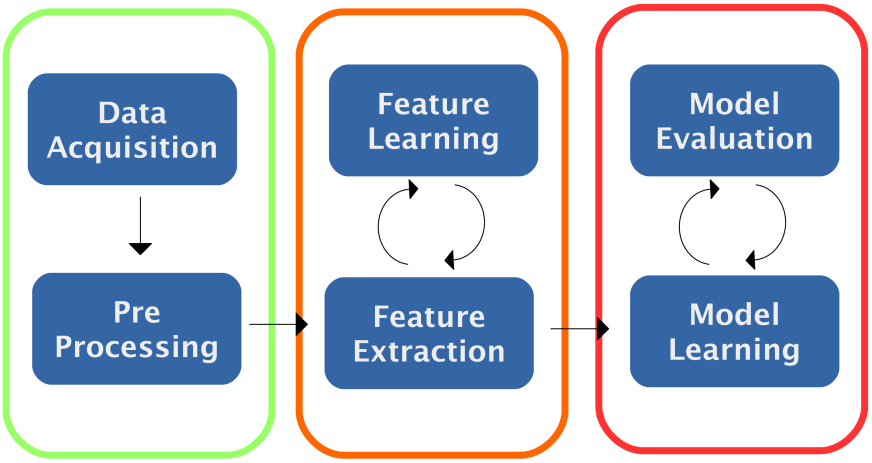


Figure 2.1: Pattern recognition general work flow.

The above problem is general for many different application domains and types of samples, such as text in a spam detector, photos in a face recognizer, and audio in a voice transcription application. By estimating $\mathcal{X}$ and M from $\mathcal{Z}_1$ by a supervised machine learning method, the label $L(s)$ assigned to new samples $s \in \mathcal{Z} \backslash \mathcal{Z}_1$ is expected to be equal to $\Omega(s)$. Some methods may count on another set $\mathcal{Z}_2 \subset \mathcal{Z}$, such that $\mathcal{Z}_1 \cap \mathcal{Z}_2 = \emptyset$, named *evaluation set*, to measure the performance of $\mathcal{X}$ and M during their design.

When the assignment errors of M are significantly lower in the *test set* $\mathcal{Z}_3 = \mathcal{Z} \backslash \mathcal{Z}_1 \cup \mathcal{Z}_2$, we say M generalizes to unseen samples. Otherwise, M is said to overfit the training data. In architecture learning, the use of $\mathcal{Z}_2$ makes the overfit less prone [59]. Note, however, that M strongly depends on the success of $\mathcal{X}$. This makes the feature learning process more important than the design of the classifier. At the same time, Deep Learning methods usually produce high-dimensional feature vectors, making the choice of M restricted to a linear classifier, such as logistic regression, support vector machines, or a stacking of linear classifiers, such as multi-layer perceptrons, that essentially transforms the feature space, specializing the neurons (features) for each category in order to apply the classification at the last layer [36, 74].

In this work, all samples derive from images. An image $\hat{I} = (D_I, \vec{I})$ is a pair, where $D_I \subset \mathbb{R}^m$, where m corresponds to the number of bands, is the image domain, and each element $p \in D_I$ is assigned to a vector $\vec{I}(p) \in \mathbb{R}^m$ of image properties, such as intensity, color components, or other measures obtained by some image transformation (e.g., linear filtering).

The set $\mathcal{Z}$ can then be composed of images or other type of sample previously isolated from them, such as pixels, superpixels (connected regions that transmit the same color and texture visual sensation), objects (connected regions with known shape), and subimages (regions of interest around objects), as illustrated in Figure 1.1. In

this work, we are interested in feature learning methods as the crucial solution of the presented problem when samples are images, subimages, objects, or superpixels.

2.2 Image feature learning

In Chapter 1 we divided the feature learning methods in two categories: knowledge-based approaches [4] and data-driven approaches [10, 23, 83, 92]. The first creates handcrafted features based on the knowledge of experts on the problem, which makes the solutions specific and difficult to scale up to different domains [11]. This turns our interest towards data-driven approaches. Among the most well succeeded paradigms, we can highlight **Dictionary Learning** [83, 92] and **Deep Learning** [10, 23]. The techniques in Dictionary Learning are usually filters rather than wrappers, since the design of the dictionary does not usually use the classification performance in $\mathcal{Z}_2$ as a criterion for optimization. These techniques are also unsupervised, usually, since the knowledge about the labels of the training samples is ignored in most of them. Although, supervised methods for dictionary learning can be found [57]. As a result, **Dictionary Learning** usually creates mid-level features rather than high-level features. By stacking multiple hidden layers, **Deep Learning** techniques can transform the low-level features, in our case represented by the pixel colors, into high-level features, which are more suitable for pattern recognition. The next sections presents both paradigms.

2.2.1 Dictionary learning

A *dictionary*, originally developed for text categorization, is composed by a collection of *words* from a *vocabulary* (*codebook*). In the case of images, we are interested in *Visual Dictionary Learning* — methods that build a collection of the most frequent image patches (words), as represented by local image features, from a set of training images, wherein all possible patches form the vocabulary. The origin of visual dictionary learning comes from the seminal article of Zellig Harris [35] from linguistics. In the general context of information retrieval, the idea is to build a *Bag of Words* (BoW) of a given vocabulary and then represent a text (sometimes related to image indexing) by its most frequent words — i.e., a feature vector of the text as a histogram of the words in the dictionary — for the purpose of text retrieval or classification[85]. In the context of content-based image retrieval, the idea was to build a *Bag of Visual Words* (BoVW) and represent an image by the histogram of its most frequent patches in the dictionary[63]. This essentially creates a mid-level image representation from low-level features, suitable for content- based image retrieval and classification problems. The patches resulting from images of different categories are expected to differ, such that histograms of images from the same category will be similar among them and dissimilar from the histograms of remaining categories.

The main operations in constructing a BoVW (also called *visual vocabulary*) from training images are [92].

i. **Interesting point detection** — detection of corresponding points (also called local features) [1] that frequently occur in the training images, such as the object corners. Methods based on the Harris-Laplace, Difference-of-Gaussian (DoG), and Hessian-Laplace operators are examples of interesting point detectors [60]. The success of these methods is questionable and some works adopt a dense/random sampling of points (see examples in Figure 2.2).

ii. **Local image description** — extraction of local image features (also called local descriptors) in patches centered at the detected points. These descriptors usually aim at being invariant to image rotation and scaling. Examples are SIFT [55] and SURF [8].

iii. **Visual world generation/vector quantization** — local descriptors extracted from all training images are grouped, usually by the k-means algorithm [26], and the representative descriptors of the groups are chosen as visual words for the dictionary.

In order to create a feature vector for a new image (also called a *term vector*), the local image descriptors are extracted from the detected interesting points and matched with the visual words of the dictionary. At this stage, there are two most popular approaches, namely *hard* assignment and *soft* assignment (and also called *pooling* approaches) [6, 14]. In hard assignment, each descriptor is matched with its closest word according to a distance function and the histogram of the words form the resulting feature vector. In soft assignment, the feature vector contains the distances between each local descriptor and all words in the dictionary.

Figure 2.3 summarizes the BoVW approach (also known as *Bag-of-Features*, BoF, approach).

2.2.2 Deep learning

Deep Learning techniques can be seen as an evolution of visual dictionaries. By stacking multiple hidden layers of neurons, they can transform local image features into mid-level image representations at the output of the initial (shallowest) layers and subsequently into high-level image representations at the output of the deepest layers. Besides the linear operation between the input image data and each neuron, other non-linear transformations, such as *activation*, *pooling*, and *normalization* can also

[1] We should avoid confusion here, since these features are not measures but image coordinates.

Figure 2.2: DoG, Harris-Laplace, and dense sampling for interesting point detection, respectively.

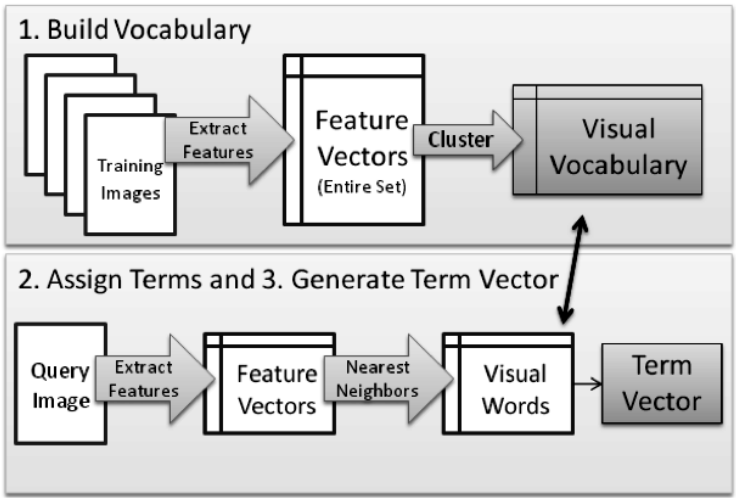


Figure 2.3: Process for BoVW image representation [63].

be applied. In the context of machine learning, these techniques can be unsupervised (generative), supervised (discriminative), or semi-supervised (hybrid) [23]. While the generative models, such as Deep Autoencoders [39] and Deep Belief Networks (DBN) [82], assume that, after unsupervised training, the parameters of the network will be able to generate dissimilar high-level representations for new images of distinct categories, the discriminative models, such as Recurrent Neural Networks (RNN) [17] and Convolutional Networks (ConvNets) [18, 42, 48], use the knowledge of the training image categories to specialize the parameters of the network such that an additional decision layer can easily assign new images to their categories. The discriminative models then usually learn features and train a classifier at the same time, as a single operation. A hybrid method can simply use the image representation from a generative network architecture as input for training a discriminative network architecture.

Perceptron and Neural Networks

Altough there is no consensus, a neural network can be considered deep when it contains more than two hidden layers, in addition to the input and output layers.

Hinton et al. [41] proposed a deep network, considering only three fully connected hidden layers.

The connection between neurons of different layers is called *synapses* (resembling the brain). Each of these neurons constitute the most important structure in a neural network, which are also known as *perceptrons* [77].

A perceptron (Figure 2.4) performs a linear operation (inner product) on the input data, followed by bias and a non-linear activation function. Considering $\vec{w} \in \mathbb{R}^k$ the synaptic weights, $b \in \mathbb{R}$ the bias, and $\varphi\colon \mathbb{R} \to \mathbb{R}$ the activation function. Given an input $\vec{x} \in \mathbb{R}^k$, the perceptron transformation is defined by

$$\varphi(\langle \vec{w}, \vec{x} \rangle + b). \tag{2.1}$$

By adjusting the weights in $\vec{w}$ and the bias b, the response of the perceptron can be adapted to different problems.

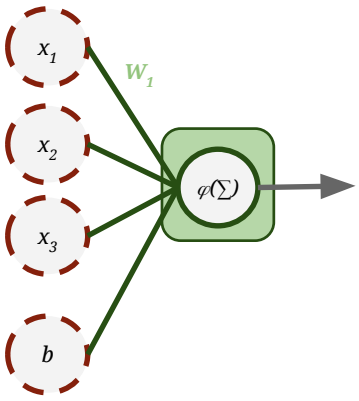


Figure 2.4: The perceptron transformation on input data.

These perceptrons encode the information with the intensity of the activation in the output $\varphi(\langle \vec{w}, \vec{x} \rangle + b)$ of all perceptrons. Such information can be encoded considering three main approaches [30]:

i. *Local Coding*: where each category is represented by a unique perceptron. Thus requiring a dimensional space equivalent to the number of categories. Despite its poor representation power, such encoding provides an easy classification, sufficing a simple $argmax$ function,

ii. *Dense Coding*: where each category is represented by the output of all perceptrons combined, which increases the representation power of the descriptor. Considering only a binary output for each neuron, such encoding strategy can represent up to 2^n different categories, and even more

when considering a real valued perceptron. However, such encoding requires a more robust classification,

iii. *Sparse Coding*: poses as an intermediate alternative between the representation power of dense encoding and the discrimination power of local encoding. Such encoding allows the activation of a small group of perceptrons per category. Considerably increasing the representation power of the descriptor, while retaining most of its discrimination power, where a simple linear classifier can be learned.

One perceptron per category can be used to make decisions in classification problems (e.g., logistic regression) [36]. Support vector machines also adopt an approach similar to the perceptron, excluding the activation function, where their parameters are learned by convex optimization [21]. Both approaches require a high-dimensional and sparse feature space, in which the categories are more likely to be linearly separable. When this is not the case, a Multi-Layer Perceptron (MLP) network can handle feature space reduction by one or more hidden layers of perceptrons followed by the output (decision) layer, as illustrated in Figure 2.5.

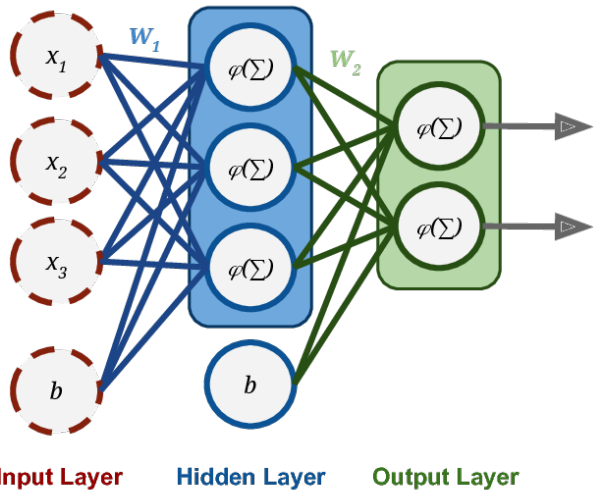


Figure 2.5: A multi-layer perceptron with three values in the input layer, three perceptrons in the single hidden layer, and two perceptrons in the output (decision) layer.

In both, logistic regression and MLP, a classifier can be trained by finding the best weight and bias sets of the perceptrons at each layer, which minimize the classification errors in $\mathcal{Z}_1$. The evaluation set $\mathcal{Z}_2$ can be alternatively used to find the best network architecture (i.e., number of hidden layers and number of perceptrons per layer). Among the possible optimization techniques, the most commonly used is the gradient descent method [36]. In the case of multiple layers, it consists of subsequent iterations of forward and backward passes. In the forward pass, the feature vector of a training

sample can be submitted to the network, with some parameter initialization, to measure the classification error and, in the backward pass, the parameters of the network can be adjusted to reduce error. This is known as the *backpropagation* algorithm, which is addressed next.

Backpropagation

For a brief introduction on the backpropagation concept, consider a neural network with layers $l = 0, 1, 2, \ldots, L$, such that $l = 0$ and $l = L$ refer to the input and output layers, respectively. We call x_i^{l-1} the output of the i^{th} perceptron (node) from the previous layer $l-1$, $l \in [1, L]$ (actually, it is the i^{th} feature input when $l-1 = 0$), and $w_{i,j}^l$ the synaptic weight of the connection between that perceptron and the j^{th} node of layer l, being b_j^l its bias term. The output x_j^l of the j^{th} node of layer l is given by

$$x_j^l = \varphi\left(b_j^l + \sum_i w_{i,j}^l \, x_i^{l-1}\right) = \varphi(o_j^l) \tag{2.2}$$

for all nodes j in layer l.

For pattern classification, the last layer $l = L$ contains as many perceptrons as the number c of categories. Its output $X^L \in \mathbb{R}^c$ is a column matrix, whose values fall within $[0, 1]$, such that the highest value indicates the chosen category for the input data $X^0 \in \mathbb{R}^n$ of the network, where n is the number of features. Let $\Lambda^c = [\lambda_1, \lambda_2, \ldots, \lambda_c]^\top$ be the local encoded column matrix with $\lambda_j = 1$, $j \in [1, c]$, at the corresponding category (row) of the input X^0 and 0 elsewhere. The decision error at the output layer L is then given by

$$E = \frac{1}{2}\|X^L - \Lambda^c\|_2^2, \tag{2.3}$$

where $\|.\|$ is the Frobenius norm. This is equivalent to

$$E = \frac{1}{2}\sum_{k_L=1}^{c} \left(x_{k_L}^L - \lambda_{k_L}\right)^2. \tag{2.4}$$

The error E depends of the choice of $w_{i,j}^l, b_j^l \ \forall i, j$ and $l = 1, 2, \ldots, L$. It can be minimized by the gradient descent approach, which updates the weights and biases of the layers according to the gradient direction of E. The partial derivative $\frac{\partial E}{\partial w_{i,j}^{L-p}}$ of E according to a given synaptic weight $w_{i,j}^{L-p}$ of the layer $L-p$, $p \in [0, L-1]$, is given by

$$
\begin{aligned}
\frac{\partial E}{\partial w_{i,j}^{L-p}} &= \sum_{k_L} \left(x_{k_L}^L - \lambda_{k_L}\right) \frac{\partial \left(x_{k_L}^L - \lambda_{k_L}\right)}{\partial w_{i,j}^{L-p}} \\
\frac{\partial E}{\partial w_{i,j}^{L-p}} &= \sum_{k_L} \left(x_{k_L}^L - \lambda_{k_L}\right) \frac{\partial x_{k_L}^L}{\partial w_{i,j}^{L-p}} \\
\frac{\partial E}{\partial w_{i,j}^{L-p}} &= \sum_{k_L} \left(x_{k_L}^L - \lambda_{k_L}\right) \frac{\partial \varphi\left(o_{k_L}^L\right)}{\partial o_{k_L}^L} \frac{\partial o_{k_L}^L}{\partial w_{i,j}^{L-p}} \\
\frac{\partial E}{\partial w_{i,j}^{L-p}} &= \sum_{k_L} \left(x_{k_L}^L - \lambda_{k_L}\right) \varphi'\left(o_{k_L}^L\right) \frac{\partial \left(b_{k_L}^L + \sum_{k_{L-1}} w_{k_{L-1},k_L}^L x_{k_{L-1}}^{L-1}\right)}{\partial w_{i,j}^{L-p}} \\
\frac{\partial E}{\partial w_{i,j}^{L-p}} &= \sum_{k_L} \left(x_{k_L}^L - \lambda_{k_L}\right) \varphi'\left(o_{k_L}^L\right) \left(\sum_{k_{L-1}} w_{k_{L-1},k_L}^L \frac{\partial x_{k_{L-1}}^{L-1}}{\partial w_{i,j}^{L-p}}\right) \\
\frac{\partial E}{\partial w_{i,j}^{L-p}} &= \sum_{k_L} \left(x_{k_L}^L - \lambda_{k_L}\right) \varphi'\left(o_{k_L}^L\right) \left(\sum_{k_{L-1}} w_{k_{L-1},k_L}^L \varphi'\left(o_{k_{L-1}}^{L-1}\right) \left(\sum_{k_{L-2}} w_{k_{L-2},k_{L-1}}^{L-1} \frac{\partial x_{k_{L-2}}^{L-2}}{\partial w_{i,j}^{L-p}}\right)\right)
\end{aligned}
\tag{2.5}
$$

and the term $\frac{\partial x_{k_{L-2}}^{L-2}}{\partial w_{i,j}^{L-p}}$ can be further expanded until $\frac{\partial x_{k_{L-p}}^{L-p}}{\partial w_{i,j}^{L-p}}$. Let $\delta_{k_{L-q}}^{L-q}$ be defined for $q = 0, 1, \ldots, p$ as follows.

$$
\delta_{k_{L-q}}^{L-q} = \begin{cases} \left(x_{k_L}^L - \lambda_{k_L}\right) & \text{if } q = 0, \text{ and} \\ w_{k_{L-q},k_{L-q+1}}^{L-q+1} & \text{otherwise.} \end{cases} \tag{2.6}
$$

Note that, $\frac{\partial x_{k_{L-p}}^{L-p}}{\partial w_{i,j}^{L-p}}$ is $\varphi'\left(o_{k_{L-p}}^{L-p}\right) x_i^{L-p-1}$ where $k_{L-p} = j$ and $k_{L-p-1} = i$. Note also that, for $p = q$, $\sum_{k_{L-p}} \delta_{k_{L-p}}^{L-p} \varphi'\left(o_{k_{L-p}}^{L-p}\right) = \delta_j^{L-p} \varphi'\left(o_j^{L-p}\right)$, since the weight $w_{i,j}^{L-p}$ affects only the output of the j^{th} neuron. Given that, Equation 2.5 can be simplified to

$$
\frac{\partial E}{\partial w_{i,j}^{L-p}} = x_i^{L-p-1} \delta_j^{L-p} \varphi'\left(o_j^{L-p}\right) \prod_{q=0}^{p-1} \left(\sum_{k_{L-q}} \delta_{k_{L-q}}^{L-q} \varphi'\left(o_{k_{L-q}}^{L-q}\right)\right), \tag{2.7}
$$

for $p = 0, 1, \ldots, L-1$, where for activation based on a sigmoid function, for example, $\varphi' = \varphi(1 - \varphi)$.

The backpropagation algorithm executes in several iterations until some convergence criterion is reached (e.g., the mean error is less than a threshold, or a maximum number of epochs is reached). After a forward pass at iteration t, the weight update of

each layer $p = 0, 1, \ldots, L-1$, is computed by

$$w_{i,j}^{L-p\,(t+1)} = w_{i,j}^{L-p\,(t)} - \alpha \frac{\partial E}{\partial w_{i,j}^{L-p\,(t)}}, \tag{2.8}$$

where $\alpha \in [0, 1]$ is a learning rate, $w_{i,j}^{l\ (t)}$ is the current weight, and $w_{i,j}^{l\ (t+1)}$ is the weight matrix to be considered in the next iteration. Similarly, by computing $\frac{\partial E}{\partial b_j^{L-p}}$, for $p = 0, 1, \ldots, L-1$, one can obtain an update equation for the bias b_j^{L-p}.

The error function (Equation 2.3) may take into consideration the prediction error of each training sample per iteration (*stochastic gradient descent*), as described above, or it can be reformulated to consider the total prediction error of all training samples per iteration — the *batch gradient descent*, which is less susceptible to noise in gradient, but also more prone to become trapped in local minima. An intermediate alternative is to consider a small number of samples in the weights update, this approach is known as *minibatch gradient descent*, where a minibatch of 1 is equivalent to the stochastic approach, while a minibatch as big as $|\mathcal{Z}_1|$ corresponds to the batch approach. Regardless the batch approach, the process of forward and backward passes for all training samples characterize one *epoch*.

Many epochs may be needed to achieve convergence, and the learning usually decreases over iterations, as the method approaches the minimum. The *learning rate decay* γ updates the learning rate as an $\alpha^{(d)}$ that decreases the original learning rate $\alpha^{(o)}$ over iterations. A common approach is the step learning rate decay, defined as

$$\alpha^{(d)} = \alpha^{(o)} \gamma^{(d)}, \tag{2.9}$$

where the iterations d are defined according to the *learning rate drop*, which consists of the number of epochs, before the learning rate is updated.

The weight (and bias) matrix updates in Equation 2.8 can also consider $\frac{\partial E}{\partial w_{i,j}^{l\ (t)}}$ of a previous iteration, introducing an inertial factor μ (*momentum*) to the learning process, which is empirically known to speed up convergence, and avoid local minima.

Finally, in order to avoid overfitting, as the synaptic weights grow, the *weight decay* λ can be considered as a regularization function over the network error E, which is equivalent to subtract a factor of the current weight in the backpropagation update.

Considering the hyperparameters *momentum*, *learning rate decay* and *weight decay*. Equation 2.8 becomes as following

$$w_{i,j}^{l\ (t+1)} = w_{i,j}^{l\ (t)} - \alpha^{(d)} \frac{\partial E}{\partial w_{i,j}^{l\ (t)}} - \mu \frac{\partial E}{\partial w_{i,j}^{l\ (t-1)}} - \lambda w_{i,j}^{l\ (t)}. \tag{2.10}$$

By referring Deep learning, we include a whole class of machine learning techniques, where many layers of information processing stages in hierarchical architectures are

exploited for both, feature learning and pattern classification [23]. Neural networks are, in the long run, a collection of perceptrons, and their variants, with different architectures and weight learning strategies.

2.3 Unsupervised neural networks

Although the scope of this work relies on supervised feature learning, and more specifically in convolution networks, unsupervised neural networks can also be used for feature learning as well, and those features may also be used for semi-supervised learning, as input to train a supervised network. This strategy might be interesting under supervised data constraint. Two basic and popular examples of unsupervised neural networks are autoencoders and the Boltzmann machines.

An autoencoder [78] is a network aiming to reconstruct in the output the same data used for input. It can have one or multiple hidden layers and the features may be represented, for instance, by the output of the most central hidden layer [7]. It might sound illogical to learn an identity function. However, when applying constraints in the representation, such as a reduced number of hidden units or sparsity constraints [62], the autoencoder is able to learn a simplified model that describes the most significant information in the input data. When adopting linear activation functions, or a single hidden layer with sigmoidal activation function, the autoencoder is strongly related to Principal Component Analysis (PCA) [15].

A Boltzman machine [40], and more specifically a Restricted Boltzmann Machine (RBM) [86], consists of one input and one hidden layer, connected with symmetrical binary synapses, assuming two different states (on and off). The synaptic weights of a RBM are learned through a Gibbs sampling approach, where the difference between the probability distribution of the input and the hidden layer output is minimized. It can be used to learn latent factors. For example, people provide a collection of movies they like and dislike, and given that those movies may be characterized as scientific fiction or fantasy, as latent factors, the network learns which type they like mostly based on the probability distribution of the latent factors.

Popular deep learning approaches, such as Stacked Denoising Autoencoders [94] and Deep Belief Networks [82], use as building blocks, the concepts drawn by autoencoders and RBMs.

2.4 Convolutional networks

Since 2012, the Convolutional Networks (ConvNets) have shown to outperform the state-of-the-art methods in the ImageNet challenge [46], being successfully applied to many scenarios. Therefore, we have chosen to focus our investigation in supervised image feature learning on ConvNets.

Each layer in a ConvNet consists of at most four operations: *convolution*, *activation*, *pooling*, and *normalization*, which require the concept of *adjacency relation* and *convolution kernel*. Differently from a MLP, where the layers are usually *fully connected* — i.e., the input data, coming from the previous layer, to each perceptron of the current layer is fixed —, the convolutional layer uses the values of a sliding window of adjacent pixels as input to the neural unit in the current layer. In addition to that, the weight set (kernel coefficients) is the same for all adjacencies at the current layer.

ConvNets consider a box adjacency relations $\mathcal{A} \subset D_I \times D_I$ between pixels of an input image $\hat{I} = (D_I, \vec{I})$. Let $p = (x_p, y_p)$ be the 2D image coordinates of a pixel p, a pixel q is said adjacent to a pixel p when

$$\mathcal{A}\colon q \in \mathcal{A}(p) \text{ if } |x_q - x_p| \leqslant \tfrac{b}{2} \text{ and } |y_q - y_p| \leqslant \tfrac{b}{2}, \tag{2.11}$$

for some kernel size $b \geqslant 3$. A convolution kernel $\hat{K} = (\mathcal{A}, K)$ is a pair where $K(q-p)$, $\forall q \in \mathcal{A}(p)$, is the weight of each adjacent pixel q. We may interpret $\mathcal{A}(p)$ as the receptive field of a neuron unit at pixel p and the values of its adjacent pixels as the input data for the neuron p. The kernel coefficients may also be vectors in a multi band kernel. In this case, the convolution kernel $\hat{K}$ is given by $(\mathcal{A}, \vec{K})$, where $\vec{K} = (K_1, K_2, \ldots, K_m)$. Therefore, the main operations in a ConvNet layer may be described as follows.

First, the input image may be normalized by

$$I'_j(p) = \frac{I_j(p)}{\sqrt{\sum_{j=1}^{m} \sum_{\forall q \in \mathcal{A}(p)} I_j(q) I_j(q)}}, \tag{2.12}$$

where $\hat{I}' = (D_I, \vec{I}')$, $\vec{I}' = (I'_1, I'_2, \ldots, I'_m)$, in order to avoid any prevalence of some pixel values over the others. Second, by using another box adjacency relation $\mathcal{B}$, the input image is convolved with a kernel bank $(\mathcal{B}, \vec{K}_i)$, $i = 1, 2, \ldots, n_k$, with n_k multi band kernels $\vec{K}_i = (K_{1,i}, K_{2,i}, \ldots, K_{m,1})$. The convolution operation with each kernel in the bank results a new image band J_i, $i = 1, 2, \ldots, n_k$, of the output image $\hat{J} = (D_J, \vec{J})$, where $\vec{J} = (J_1, J_2, \ldots, J_{n_k})$:

$$J_i(p) = \sum_{\forall q \in \mathcal{B}} \langle \vec{I}'(q), \vec{K}_i(p-q) \rangle. \tag{2.13}$$

The values $J_i(p)$ are subsequently submitted to an activation function φ. Considering $J_i(p) \leftarrow \varphi(J_i(p))$, the features extracted with convolution must be aggregated by *pooling*, which is intended to cope with possible object shifts in the training images. The pooling operation takes into account another adjacency relation $\mathcal{C}$ and outputs an

image $\hat{U} = (D_U, \vec{U})$, where $\vec{U} = (U_1, U_2, \ldots, U_{n_k})$ and

$$U_i(p) = \sqrt[\beta]{\sum_{\forall q \in \mathcal{C}(p)} J_i(q)^\beta}. \tag{2.14}$$

The value β controls the operation which can varies from *additive* ($\beta = 1$) to *max-pooling* (β is too high, e.g., $\beta = 10$). The pooling can also reduce the dimension $|D_U| = \frac{|D_J|}{s^2}$ by a *stride* factor $s \geqslant 1$, when it is computed only for pixels at each shift of s pixels horizontally and vertically, respectively (Figure 2.6). Finally, the feature values in $\hat{U}$ can be normalized and used as input for a next layer.

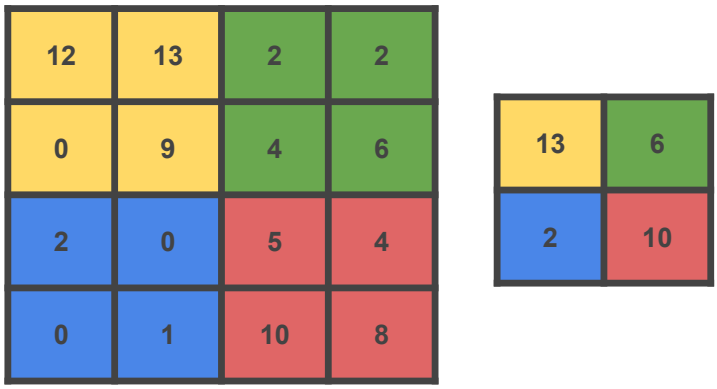


Figure 2.6: Example of a 2x2, and *stride* 2, *max-pooling* operation.

When learning the kernel weights by backpropagation, one may consider a last logistic regression layer or a MLP after the last layer of the ConvNet (Figure 2.7). In this case, the weights and biases of the MLP are learned as well, making feature learning and design of a pattern classifier a single operation. However, in very deep networks, the derivatives of the error in backpropagation can become too small, or even zero (vanish), for the first ConvNet layers. This is known as the *vanishing problem* and can be alleviated if we choose a *Relu* (Rectified linear) activation function $\varphi(p) = \max(p, 0)$ [34], which has a derivative output $\varphi'_i(p) = \min(\text{sign}(p) + 1, 1)$. Note also that the main purpose of activation is to create a sparse code [30].

By stacking several layers, the ConvNet transforms low-level into high-level image features. The output feature vectors are usually high-dimensional and sparse, so they can also be used with linear classifiers such as logistic regression and SVM (see example in Figure 2.8).

Considering the weights of a network as its *parameters*. The sizes of the above adjacency relations, number of kernels at each layer, number of layers, stride, etc., are all considered *hyperparameters*, as they define the general architecture for the network. We then identify two main strategies for supervised image feature learning with ConvNets: *Filter Learning* [51, 76] and *Architecture Learning* [12, 59, 69, 87]. The first fixes an architecture (hyperparameters) and learns the kernel weights (parameters) by backpropagation, while the second randomly chooses the weights of the kernels

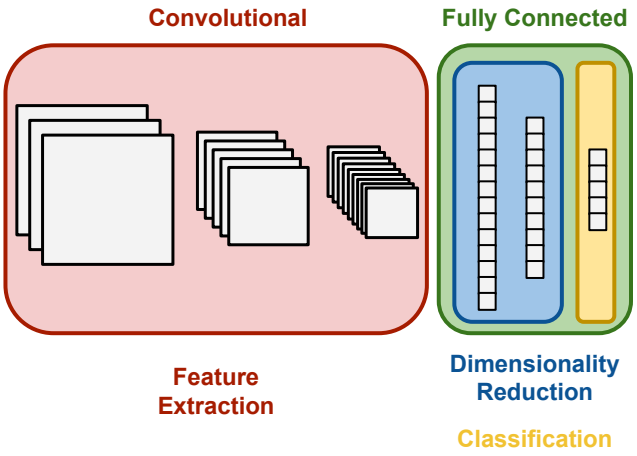


Figure 2.7: A ConvNet followed by MLP for backpropagation. The ConvNet performs feature extraction, the first fully-connected layers of the MLP perform feature space reduction, and the last layer of the MLP is responsible for the final decision.

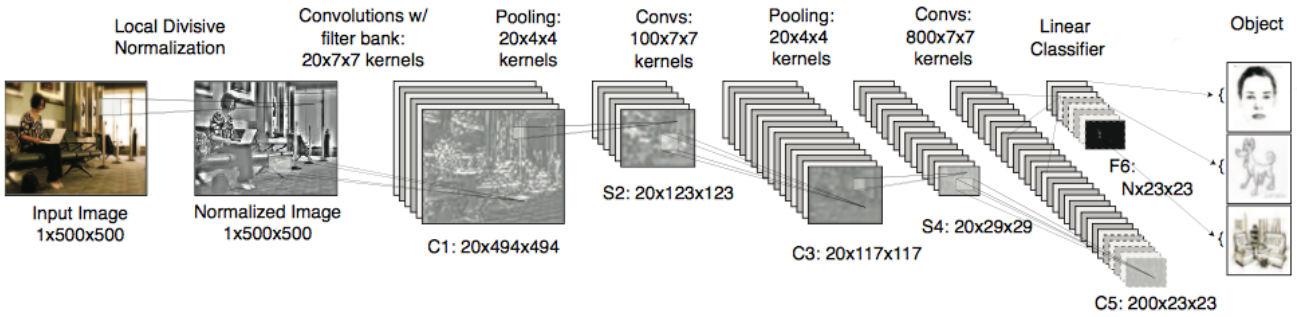


Figure 2.8: Example of ConvNet architecture for object recognition [28].

(parameters) while learning the architecture (hyperparameters) by using the evaluation set. *Transfer Learning* can also be adopted, where we adapt a previously learned network, with its *parameters* and *hyperparameters* for a different scenario.

The next chapter then proposes a methodology to study the effectiveness of image feature learning by ConvNets under supervised data constraint according to these three approaches.

Chapter 3

Methodology

"No great discovery was ever made without a bold guess."

—Isaac Newton

Deep learning techniques, and of course Convolutional Networks (ConvNets), are known for requiring large training sets to learn the parameters (weights and biases) of the network [75]. Whenever, large supervised training sets require manual labeling by experts, which constitutes a serious drawback for deep learning methods [53]. However, ConvNets architecture (using random weights) could be learned in scenarios with no huge amount of supervised data for some categories of problems, such as iris spoofing detection [59]. These facts aroused the interest in the present study about image feature learning by ConvNets under supervised data constraints.

The absence of supervised training samples in deep learning can be addressed by unsupervised and semi-supervised methods, such as Deep Belief Networks [41] and Stacked Auto Encoders [94]. The Stacked Deep Polynomial Network [84] is another example, proposed for tumor classification, which combines data-driven with handcrafted features. In this chapter, we propose to investigate ConvNets under supervised data constraint, with no help from handcrafted features and unsupervised samples from the problem. We believe both can improve our results, but we are interested in understanding the pros and cons of ConvNets when using a reasonably low number of supervised samples only.

The chapter is organized as follows. Section 3.1 describes the methods considered for ConvNet-based image feature learning. These methods must be evaluated on data sets of various distinct characteristics: type of sample, balanced and unbalanced number of samples per category, image domain, and number of categories. Section 3.2 describes the selected data sets that satisfy such a heterogeneity condition. A reasonably low number of supervised samples for the feature learning process is established for each data set. The main questions in this study and the proposed experiments to answer them are presented in Section 3.3.

3.1 Image feature learning by ConvNets

The three main approaches identified in Chapter 2 for ConvNet-based image feature learning are described in this section: Filter Learning [46, 51], Transfer Learning [25, 64, 91], and Architecture Learning [12, 59, 69, 87]. In order to compare them, we must separate the feature extraction layers from the other layers and consider their output features as input to a same pattern classification model.

3.1.1 Filter learning

By fixing the ConvNet architecture and using its last layer as the input layer of a Multi-Layer Perceptron (MLP), one can use the backpropagation algorithm (Section 2.2.2) to learn not only the parameters of the fully-connected layers of the MLP, but also the weights of the kernel banks in each layer of the ConvNet (Figure 2.7). We have chosen the architecture of the AlexNet network [46] for this purpose, since it has been trained in a large image data set. Figure 3.1 illustrates the architecture of the AlexNet. Where the input images must have 224×224 pixels and 3 color channels in the RGB color space. The ConvNet contains 5 layers, C_i, $i = 1, 2, \ldots, 5$. The kernel bank of the first layer contains 96 kernels of $11 \times 11 \times 3$ coefficients. The activation function is Relu and is applied after convolution with the kernel bank of each layer, as well as in the fully connected layers. However, normalization N_i followed by max-pooling P_i is used only in layers C_i, $i = 1, 2$, and max-pooling is also applied in C_5. Due to the strides and number of kernels, the spatial resolutions of the output multi band images of C_i, $i = 1, 2, \ldots, 5$, are $55 \times 55 \times 96$, $27 \times 27 \times 256$, $13 \times 13 \times 384$, $13 \times 13 \times 384$, and $6 \times 6 \times 256$, respectively. Therefore, the output feature vector of the ConvNet contains $6 \times 6 \times 256 = 9,216$ features (input layer of the MLP). And the two fully-connected hidden layers of the MLP reduce the input feature vector from $9,216$ to $4,096$ features, and the decision layer outputs the final vector with the likelihood values of the $1,000$ categories.

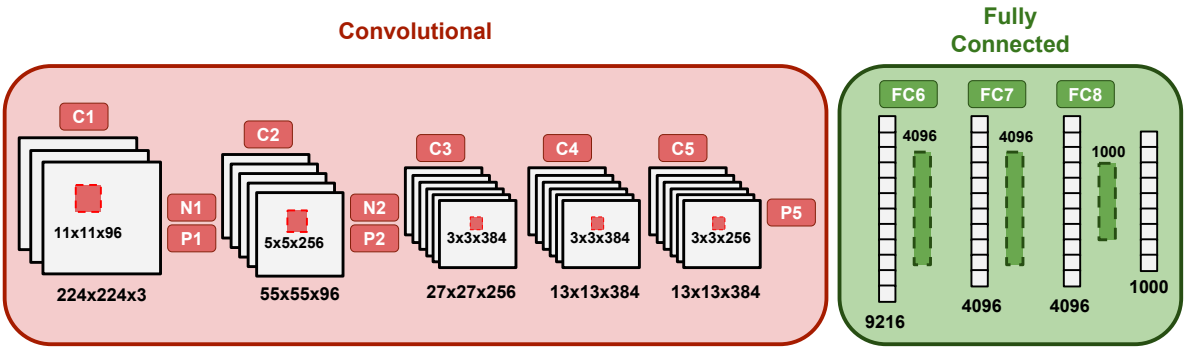


Figure 3.1: An illustration of AlexNet architecture. Denoting all Convolutional (C), Normalization (N), Max Pooling (P) and Fully Connected layers (FC), interleaved by Relu activation units.

In order to evaluate the effect of supervised data constraint in Filter Learning,

all weights in the AlexNet are randomly initialized, allowing us to verify if the FL approach can find effective parameters from a limited number of supervised training samples. In order to use the AlexNet in different data sets, all input images must be interpolated to the input dimension of the AlexNet and the output layer must be changed to contain the number of categories (classes) required by the new problem.

As previously mentioned, a fair comparison among the learning approaches require image descriptors to be fed to the same classifier without dimensionality reduction. For this reason, we use the output of P_5 as image descriptor and the classifier is an SVM — one-*versus*-all for c categories, depending on the data set, with parameters optimized by grid search (Figure 3.2).

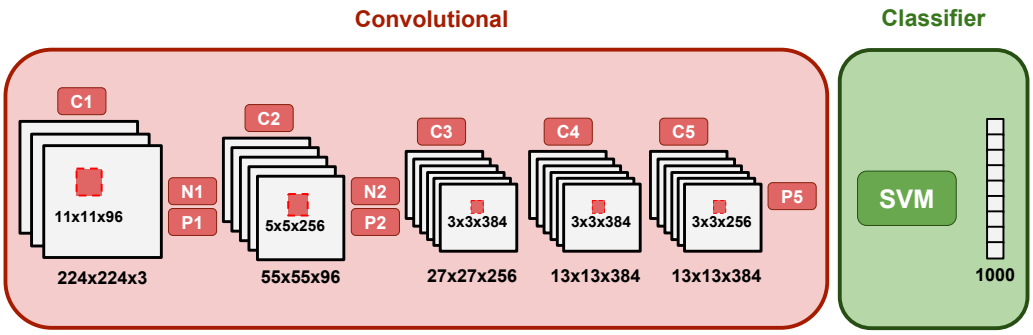


Figure 3.2: An illustration of the evaluated AlexNet descriptor. Feeding the output of the last convolutional layer to a linear SVM classifier.

In order to evaluate the behavior of such descriptors in an distance-based classifier, we also include the Optimum-Path Forest classifier [65] in our experiments, following the same approach applied in SVM.

3.1.2 Transfer learning

The AlexNet has also shown excellent performance, when its original parameters are fine tuned for a new application [64, 89, 95]. The strategy, known as Transfer Learning from source to target domains [91], is more promising to succeed than Filter Learning given a supervised data constraint [89]. The learning process starts with the original weights of the AlexNet [95]. Given that the first layers are usually more stable, we set a learning factor for the last layer 10 times higher than the factors of the previous layers in the AlexNet (Figure 3.1). After this fine tunning, the MLP is substituted by the SVM classifier as in Filter Learning.

3.1.3 Architecture learning

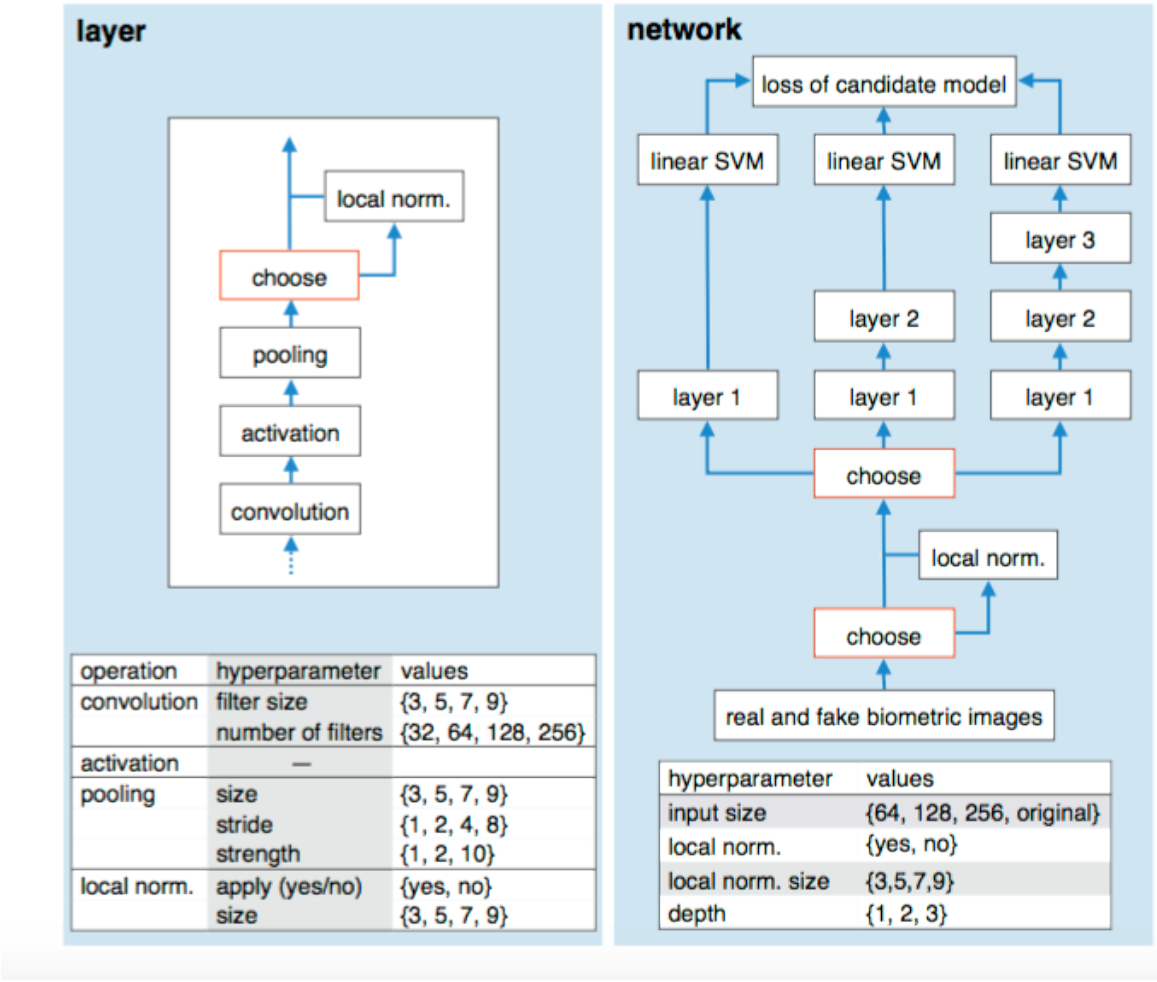
In Architecture Learning [69], the kernel weights of ConvNet are drawn from a Gaussian distribution chosen to have mean 0 and norm 1. Such a setting usually produces orthonormal kernels at each layer. The aim is to learn the hyperparameters

represented by the number of layers, kernels (and their size), the pooling factor β, as well as the window size and stride applied for pooling and normalization. The kernels, in this case, together with the Relu activation function, produce a sparse code, since 50% of the convolution results tend to be positives and the remaining 50% negatives.

Search the best set of hyperparameters can be performed by adding a last decision layer (SVM or logistic regression) to the ConvNet, using the training set to find the weights of the classifier, and evaluating the results of different hyperparameter combinations, as feature extractors, on an evaluation set. In this work, we adopt an SVM classifier evaluating a 5×2 cross validation, considering the same hyperparameter search space (Figure 3.3) used in [59], where, in a network up to 3 layers, we have 25 hyperparameters for optimization.

- Layer Parameters:
  - convolution filter size,
  - number of convolution filters,
  - pooling filter size,
  - pooling stride,
  - pooling factor,
  - apply/not normalization filter, and
  - normalization filter size.
- Network Parameters:
  - input image size,
  - apply/not input normalization, and
  - normalization filter size,
  - depth.

As shown in [59], Architecture Learning has been able to learn well suited descriptors for a iris spoofing scenario, even without huge training data sets. However, the work in [22] has shown that only a very small portion of the evaluated architectures are able to show a reasonable performance. Figure 3.4 shows the hyperparameter search space for a face recognition problem [22] and the corresponding accuracy of the evaluated models, where most evaluated architectures present a considerably poor performance.



operation	hyperparameter	values
convolution	filter size	{3, 5, 7, 9}
	number of filters	{32, 64, 128, 256}
activation	—	
pooling	size	{3, 5, 7, 9}
	stride	{1, 2, 4, 8}
	strength	{1, 2, 10}
local norm.	apply (yes/no)	{yes, no}
	size	{3, 5, 7, 9}

hyperparameter	values
input size	{64, 128, 256, original}
local norm.	{yes, no}
local norm. size	{3,5,7,9}
depth	{1, 2, 3}

Figure 3.3: Hyperparameter search space for architecture optimization [59].

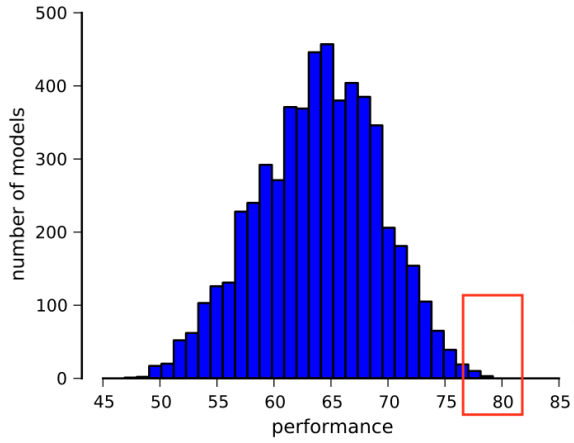


Figure 3.4: Accuracy distribution of evaluated models in Architecture Learning for face recognition [22].

3.2 Data sets

Deep Learning techniques, specially ConvNets, have achieved good classification performance in many different scenarios [31, 46, 59, 61]. In order to evaluate ConvNets under supervised data constraint, we have chosen a group of image classification data sets with diverse characteristics in respect to the number of samples per class, number of classes, sample type (image, subimage, superpixel, and object), image properties, etc. Some of these data sets have been proposed in the literature with specific protocols to evaluate image classification. However, those protocols do not represent a scenario of supervised data constraint. Hence, we have designed our own evaluation protocols for all data sets, aiming to answer the questions raised in Section 3.3.

We have chosen eight data sets, representing different types of samples: images (Stl10, Cifar10, Scenes15, and Melanoma), objects (Larvae, Mnist), subimages (Pubfig5), and superpixels (Rome). The number of categories of these data sets varies from 2 to 15, and the categories are unbalanced in number of samples for Larvae, Melanoma, and Rome. These data sets also vary in many fundamental image properties, such as color, dimension and ratio. Only Scenes15, Mnist and Pubfig5 lack in color information, being composed by grayscale images. All image dimension vary from a 28×28 to 1816×742 pixels.

In following sections we describe, in more details, the properties of each data set.

3.2.1 Stl10

Stl10 [20] consists of natural images extracted from the ImageNet challenge [79]. It contains $13,000$ images (samples) uniformly distributed in 10 distinct categories: airplane, bird, car, cat, deer, dog, horse, monkey, ship, and truck.

Different from the original ImageNet data set, all images in Stl10 have the same dimension (96×96 pixels), as the examples in Figure 3.5.

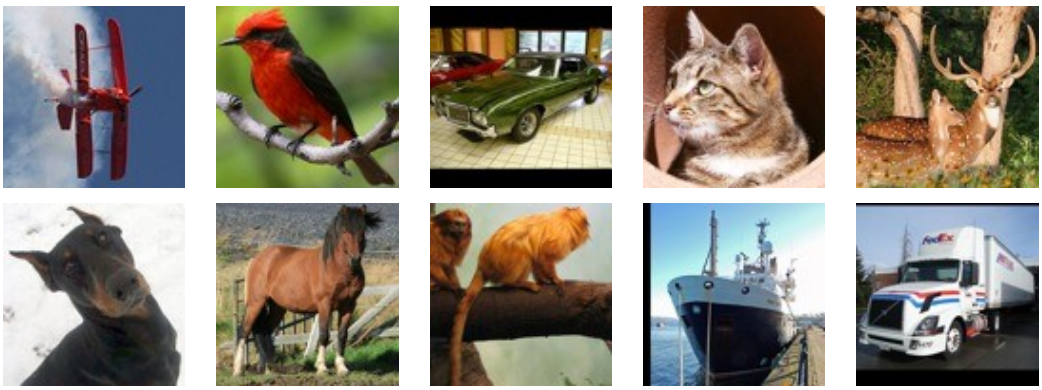


Figure 3.5: Examples of images from all categories in Stl10.

3.2.2 Cifar10

Cifar10 [45] is a data set with 10,000 natural images (samples) uniformly distributed in 10 categories: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. Although it has not been derived from ImageNet, the images in Cifar10 and Stl10 have some similarities. The images in Cifar10, however, present a considerably lower dimension of 32×32 pixels.

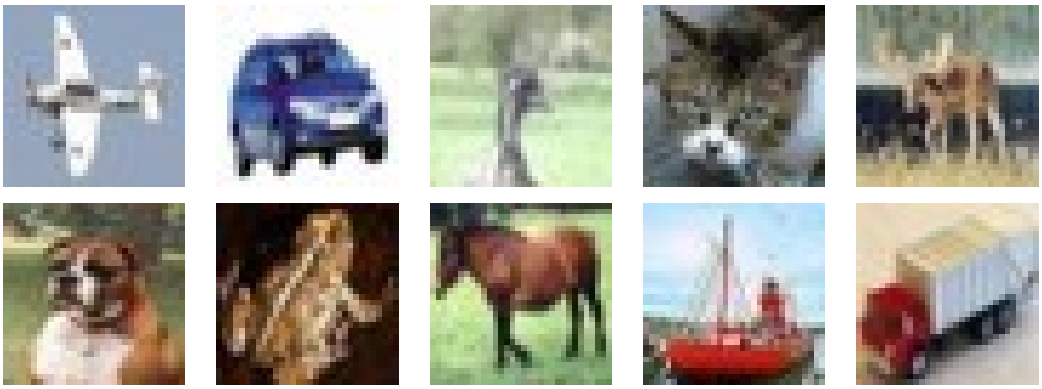


Figure 3.6: Examples of images from all categories in Cifar10.

3.2.3 Larvae

The data set Larvae is private and it has been obtained from our project to automate the diagnosis of parasites [88]. The original images are regions of interest automatically detected and extracted from 4M-pixel images of microscopy slide fields. All images have a rectangular shape of 1816×742, where a black background is added to turn them a square image, followed by a resize interpolation to 224×224 pixels. In Larvae, there are two types of categories: S. *stercoralis* (with 476 samples) and fecal impurity of similar size (with 3,068 samples). Together, they represent a data set with 3,544 images, being some impurities very similar to *larvae* in shape. Figure 3.7 shows examples where these categories are different in shape and position. A special aspect of this data set relies on the strong unbalance in number of samples. Although we have not segmented the image, samples are represented by a single segmented object (a different sample type than the previous ones).

3.2.4 Melanoma

Melanoma contains 1,039 images of skin lesions, being 495 images of benign tumors (nevus), 272 images of malign tumors (melanoma), and the remaining 272 images of other injuries. This is also a private data set, which was kindly provided by Prof. M. Emre Celebi from the Dept. of Computer Science, University of Central Arkansas. Due to different image acquisition protocols, the original images considerably change in

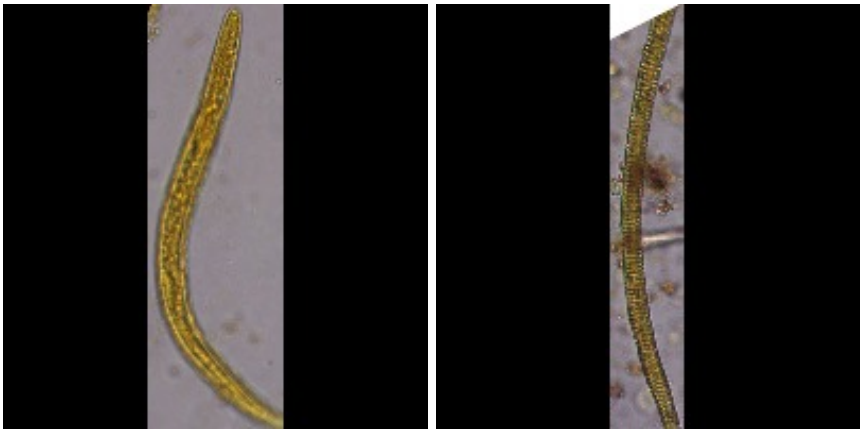


Figure 3.7: Examples of a *helminth larvae* (left) and a fecal impurity (right) from the data set Larvae.

dimension (varying from 220×203 to 220×552 pixels), illumination, focus, partial occusion by artifacts such as hair, ruler, mesh, etc. To apply on ConvNets, the images are interpolated to a 224×224 pixels(Figure 3.8). Again, the sample type is an object with some uniform background.

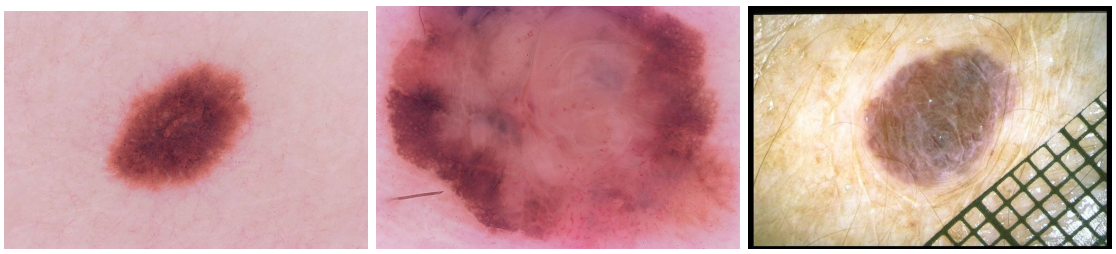


Figure 3.8: Examples of images from the three categories in Melanoma: nevus (left), melanoma (center), and other injury (right).

3.2.5 Mnist

Mnist [50] contains $60,000$ images of handwritten digits from 0 to 9, representing 10 uniformly distributed categories. The digits are represented by good-contrast grayscale images of only 28×28 pixels, being the data set with smallest dimension, among the selected ones. Again, the object (sample) is the only information in the image. Figure 3.9 shows examples extracted from the Mnist dataset.

3.2.6 Pubfig5

Pubfig5 is a data set with only 5 categories extracted from the original Pubfig [47], which contains $58,797$ face images of 200 popular people, as collected from the internet, without any pose, illumination, or alignment restrictions (i.e., the scenario is said to be *in the wild*). The faces and eyes in the images have been automatically detected, the images were aligned by the positions of the eyes, and a region of interest with 200×200

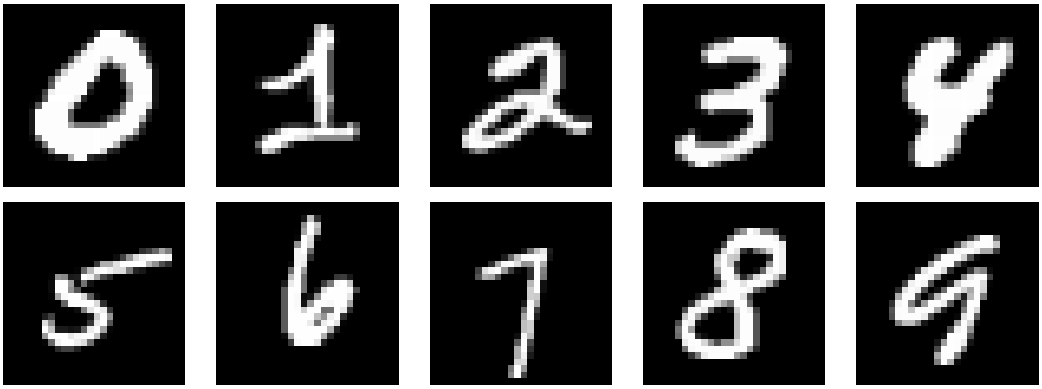


Figure 3.9: Examples of images from all categories in Mnist.

pixels was extracted around the face to create the samples of the data set. Pubfig5 was created with the five most frequent categories out of the original 83 individuals in the data set, with 299, 300, 300, 354 and 367 images, respectively, composing a roughly balanced data set with 1,620 grayscale images.

Figure 3.10: Examples of images from all categories in Pubfig5.

3.2.7 Scenes15

Scenes15 is a data set with 8,760 grayscale images of natural scenes, with dimensions varying from 203×220 to 552×220 pixels each, distributed in 15 categories [49]: office (215 samples), kitchen (210 samples), living room (289 samples), bedroom (216 samples), store (315 samples), industrial (311 samples), tall buildings (356 samples), inside city (308 samples), street (292 samples), highway (260 samples), coast (360 samples), open country (410 samples), mountain (374 samples), forest (328 samples), and suburb (241 samples), which are interpolated to the input size of the ConvNet. As shown in Figure 3.11, it can be strongly difficult to distinguish across some categories, even for humans, since their categorization depends on the contextual information not always represented by the image content.

3.2.8 Rome

Rome [93] comes from a high-resolution aerial image taken from the city of Rome. The original image contains $2,817 \times 2,847$ pixels and we have segmented it into 24,968

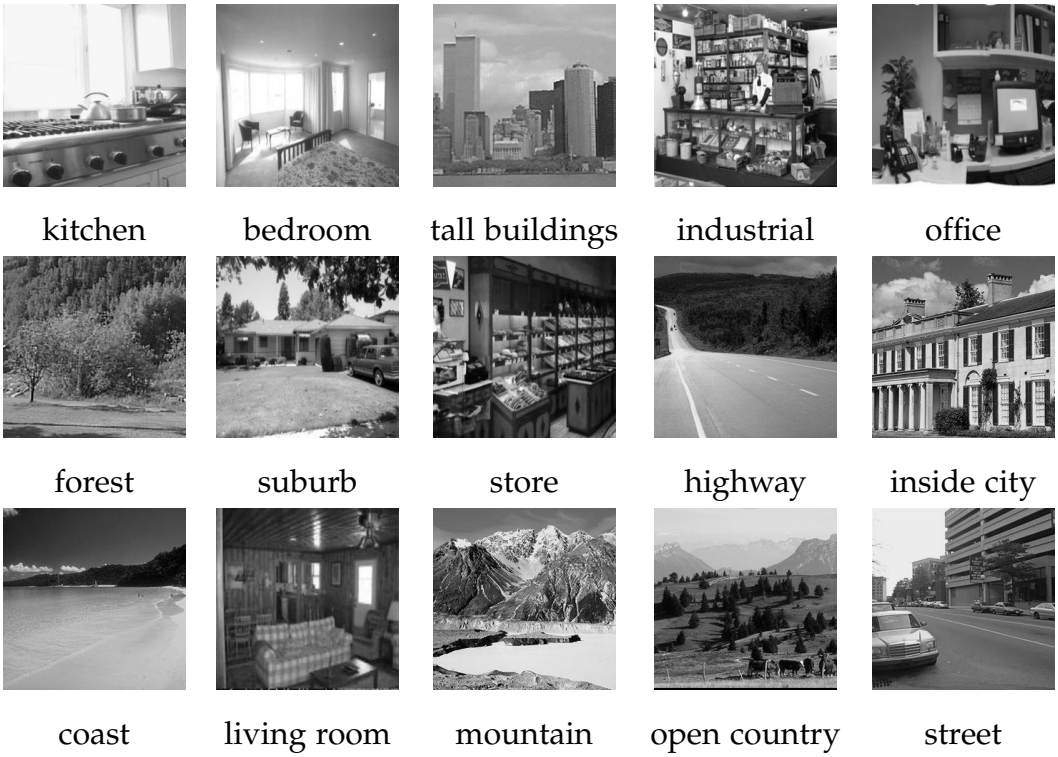


Figure 3.11: Examples of images from all categories in Scenes15.

superpixels (samples) [1], in order to evaluate the ability of ConvNets in handling small and irregular homogeneous regions. The superpixels have been pre-annotated into 7 categories: road (2048 samples), tree (2936 samples), shadow (4702 samples), water (843 samples), building (13082 samples), Grass (1021 samples) and bare soil (336 samples). For deep learning, we extracted a small region of interest around each given superpixel, showing its context, and scaled it to the dimension of the corresponding input of the ConvNet.

Figure 3.12: A portion of the aerial image of Rome, showing the obtained superpixels.

As superpixels are essentially composed of non regular areas, a pre processing stage is needed in order to solve this non regularity issue. In this work, we consider a bounding box surrounding the superpixel, with an extra padding of 10% in each dimension. This bounding box is then interpolated to match the input of the ConvNet. An illustration of the process is shown in Figure 3.13.

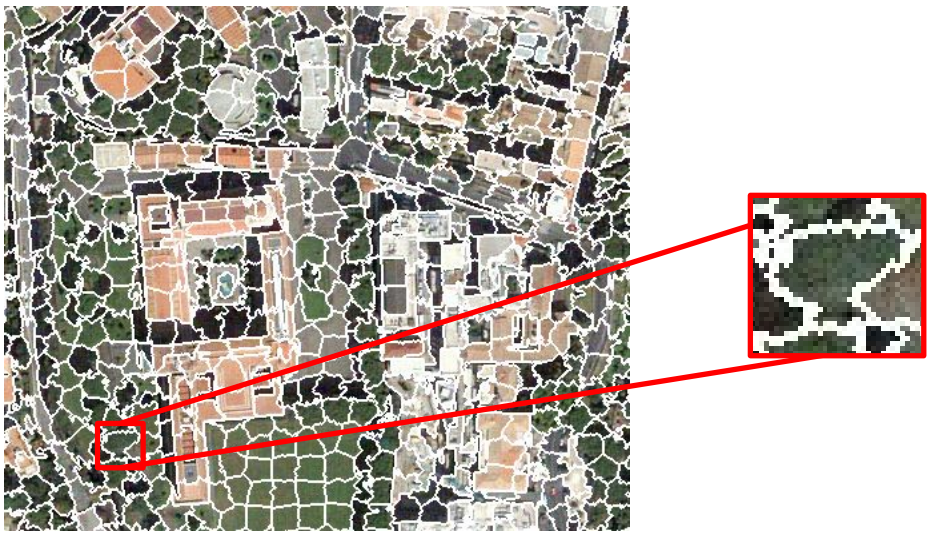


Figure 3.13: An example of the sample extraction and interpolation applied in the Rome data set.

The data sets presented in this section comprises a wide range in several aspects, including dimension, color, ratio, number of classes and balance, in an attempt to evaluate the behavior of ConvNets in many different scenarios. Table 3.1 summarizes the characteristics of all data sets considered in the following experiments:

Data sets	dimension	color	ratio	# classes	# samples	balanced
Stl10	96×96	yes	constant	10	13,000	yes
Cifar10	32×32	yes	constant	10	60,000	yes
Larvae	1816×742	yes	constant	2	3,544	no
Melanoma	$[220 \times 203, 220 \times 552]$	yes	variable	3	1,039	no
Mnist	28×28	no	constant	10	60,000	yes
Pubfig5	200×200	no	constant	5	367	roughly
Scenes15	$[203 \times 220, 552 \times 220]$	no	variable	15	8,760	no
Rome	$[10 \times 8, 54 \times 62]$	yes	variable	7	24,968	no

Table 3.1: An overview of all data sets considered in the experiments.

3.3 Research questions and proposed experiments

In order to assess the performance of Feature Learning (FL), Transfer Learning (TL), and Architecture Learning (AL) for supervised data constraint, we first have to define a reasonably low number of samples for manual annotation. For the presented data sets, is feasible and reasonable to expect the specialist to be able to annotate 100 images per category. In a real application, those images could be carefully selected by the specialist to better represent their categories. In this work, we simply randomly selected 100 images per category from each data set. They constitute the respective training set for image feature learning. The remaining images, for which we also know the label, have been used to compose the test set. As we will see in the next chapter, the experiments have also been repeated for different choices of training, and test sets, being the test set fixed for all approaches, in order to obtain the statistics of the results.

Under such data constraint, we want to figure out which is the best method for image feature among FL, TL, and AL? FL has a tendency to overfit, while TL can considerably reduce the overfit and AL has shown to be effective with data limitation in spoofing detection [59]. Therefore, we want to evaluate first, in the next chapter, the best among the learning approaches to deal with supervised data restrictions. However, how do we know the method with the best performance has learned an effective feature space? When the feature space is effective, we should expect the accuracy of a suitable classifier to increases with the number of supervised samples. We then propose to observe the behavior of the SVM classifier when additional samples are selected from the evaluation set and included in the training set by active learning. The underlying idea here is to use the classifier to select the most important samples for its learning process.

We believe that, using unsupervised samples from the same problem we can improve the results of classification. However, for the present study, we would like to explore only the supervised samples in the training set. This implies in the use of data augmentation by affine transformations on our training samples. Can such type of data augmentation improve image feature learning for any of the three methods, FL, TL, and AL?

Another important aspect is the ability of reducing the feature space to make feasible the use of operations that rely on distance functions, such as clustering methods, distance-based classifiers, and content-based image retrieval applications. The use of an MLP allows feature space reduction when the feature vector is extracted from the output of the last hidden layer, as well as traditional dimensionality reduction approaches, namely, PCA and LDA. Therefore, can we reduce the obtained feature space from FL, AL, and TL?

Therefore, the next chapter is dedicated to answer the above questions and the subsequent one evaluates the best solution for a real application.

Chapter 4

Experiments and Results

"No particular theory may ever be regarded as absolutely certain.... No scientific theory is sacrosanct..."

—Karl Popper

In the preceding chapters we have presented the main concepts of Deep Learning, focusing on Convolutional Networks (ConvNets), as well as the main contributions these techniques have brought into machine learning and image analysis fields. We have also raised some questions about the behavior of ConvNets in overlooked scenarios, with special attention to problems under supervised data constraint. This chapter presents the experiments designed to answer those questions and a discussion about their results.

4.1 General setup

We consider three approaches for image feature learning: Architecture Learning (AL), Filter Learning (FL), and Transfer Learning (TL). This section describes the hyperparameter architecture search space adopted for AL, as well as the backpropagation hyperparameters adopted to learn weights and biases in FL and TL.

ConvNets can be extremely expensive due to the large number of convolutions and inner products to be computed. Since these operations are easily parallelized, modern implementations of ConvNets rely on the recent advances in *Graphics Processing Units* (GPUs) to speedup both, learning and deployment of such networks. In order to benefit from these implementations, as well as make the results reproducible, all experiments are performed with Simple-HP [18], which is based in Theano [90], and Caffe [43] frameworks, for hyperparameter and parameter optimization, respectively.

For AL, we choose the best descriptor among $2,000$ randomly generated architectures, while considering the AL search space, inspired by [59], described in Table 4.2.

Notice that, in order to deal with the small images obtained with superpixels, we include an extra input size dimension of 32×32 pixels.

Table 4.1: Architecture Learning hyperparameters space.

Layer	
Convolution Filter Size	$\{3, 5, 7, 9\}$
Number of convolution filters	{32,64,128,256}
Pooling filter size	{3,5,7,9}
Pooling stride	{1,2,4,8}
Pooling factor	{1, 2, 10}
Apply/Not normalization filter	yes/no
Normalization filter size	{3,5,7,9}

Network	
Input Size	{32,64,128,256}
Apply/Not normalization filter	yes/no
Normalization filter size	{3,5,7,9}
Number of layers	{1,2,3}

For FL and TL, we adopt the approach employed in [46] with the traditional AlexNet architecture, training through 40 epochs of backpropagation. After the first 40 epochs, the original learning rate is dropped by a 0.1 decay, and followed by another 40 extra epochs. The backpropagation hyperparameters are chosen as follows:

FL and TL use the entire training set for feature learning ($\mathcal{Z}_2 = \emptyset$), while AL adopt 5×2-fold cross validation over the training samples. That is, 50% of the samples ($\mathcal{Z}_1$) are randomly chosen to train the SVM classifier for a given architecture and the other 50% are selected for the evaluation set $\mathcal{Z}_2$, used to assess the quality of the architecture (i.e., classification accuracy in $\mathcal{Z}_2$). This process swaps $\mathcal{Z}_1$ and $\mathcal{Z}_2$, and repeats 5 times to obtain the mean score of the architecture. Once the architecture with the highest mean score is chosen, all samples ($\mathcal{Z}_1 \cup \mathcal{Z}_2$) are used to train the SVM classifier for final assessment in the unseen test set $\mathcal{Z}_3$.

[1]For FL and TL respectively. The considerably smaller learning rate for TL is justified by the fact the weights in TL are expected to be slightly adjusted only.

Table 4.2: Filter and Transfer Learning stochastic gradient descent setup.

Backpropagation	
Momentum	0.9
Learning Rate	$0.01/0.0001^{1}$
Weight Decay	0.05
Learning Rate Drop	40 epochs
Learning Rate Decay	0.1
Batch Size	50

4.2 What is the best image feature learning approach?

We are seeking the best image feature learning approach, among AL, FL, and TL, when the training set contains only 100 supervised samples per category. Two challenges stem from the fact ConvNets usually require large training sets [44, 46] and suffer with category imbalance. The second problem justifies our choice for a uniform number of training samples per category [38]. It is important to point out that, the imbalance remains present in the test set of some data sets, as shown in the previous chapter.

After image feature learning with AL, FL, and TL, the output of the last ConvNet layer is used as input to project a final SVM classifier with the entire training set ($\mathcal{Z}_1 \cup \mathcal{Z}_2$). The process is repeated considering different training and test data sets, where different descriptors are learned and evaluated. The average of balanced accuracy (numerical value on bars), along with the standard deviation (error bar in black), are presented in Figure 4.1.

Considering a pairwise t-student test, TL significantly outperforms AL and FL, except for the data sets Rome, Pubfig5, and Larvae. For these data sets, TL and AL produce similar results. Indeed, whenever the texture information is higher (more heterogeneous images), the results of TL and AL approximate each other. All methods have presented a poor performance on Rome, indicating that the respective ConvNets can not extract meaningful image features from superpixels. Perhaps, it is caused by the absence of texture in those superpixels. Considering a pairwise comparison between AL and FL, the t-student test shows AL is consistently better than FL, except for Melanoma, as seen in Figure 4.2.

Another question is related to the quality of the AlexNet's architecture. Can we attribute the success of TL to the architecture of the AlexNet? This question can be answered by substituting the ConvNet's weights of the AlexNet by the randomly initialized weights, such as in AL. Figure 4.3 presents a comparison between the learned architecture and the AlexNet's architecture with random weights.

The learned architecture shows indeed better results than the AlexNet's architecture

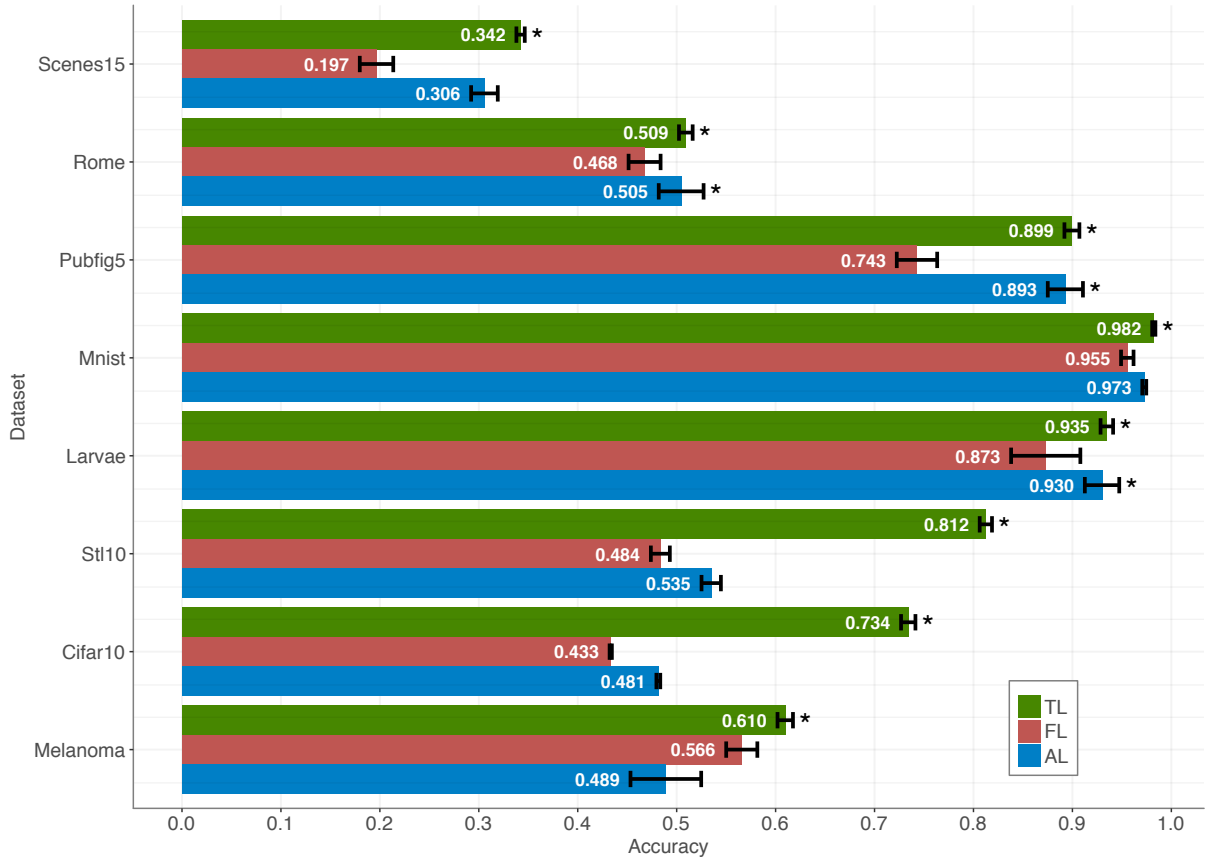


Figure 4.1: Pairwise comparison between TL and FL, and between TL and AL in accuracy of classification (∗ indicates statistical significance).

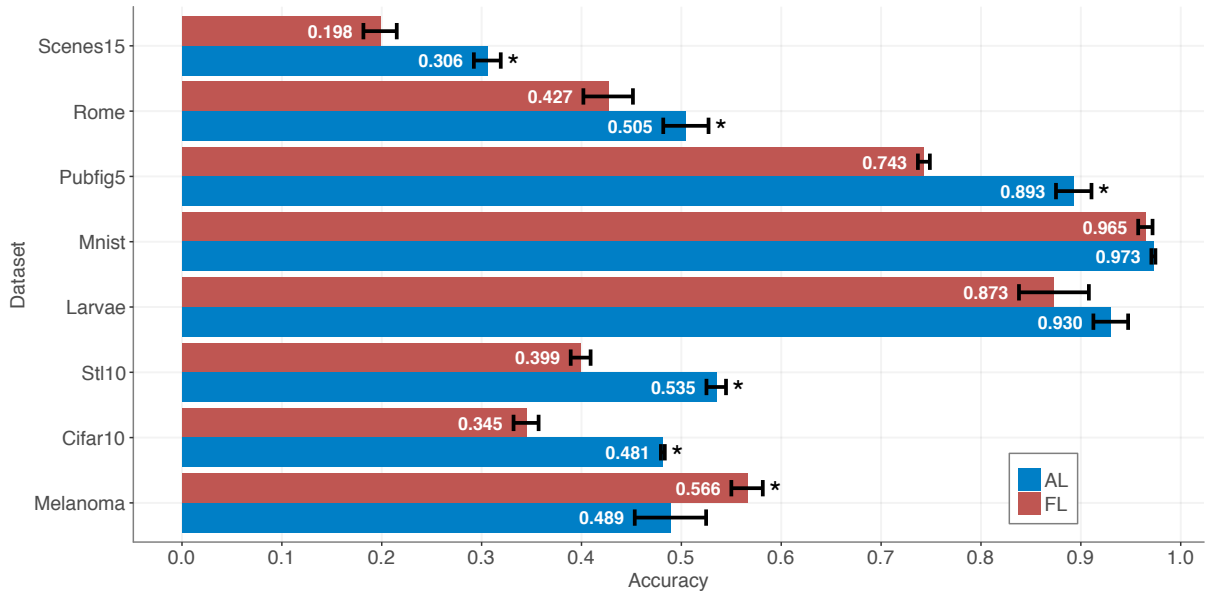


Figure 4.2: Pairwise comparison between AL and FL in accuracy of classification (∗ indicates statistical significance).

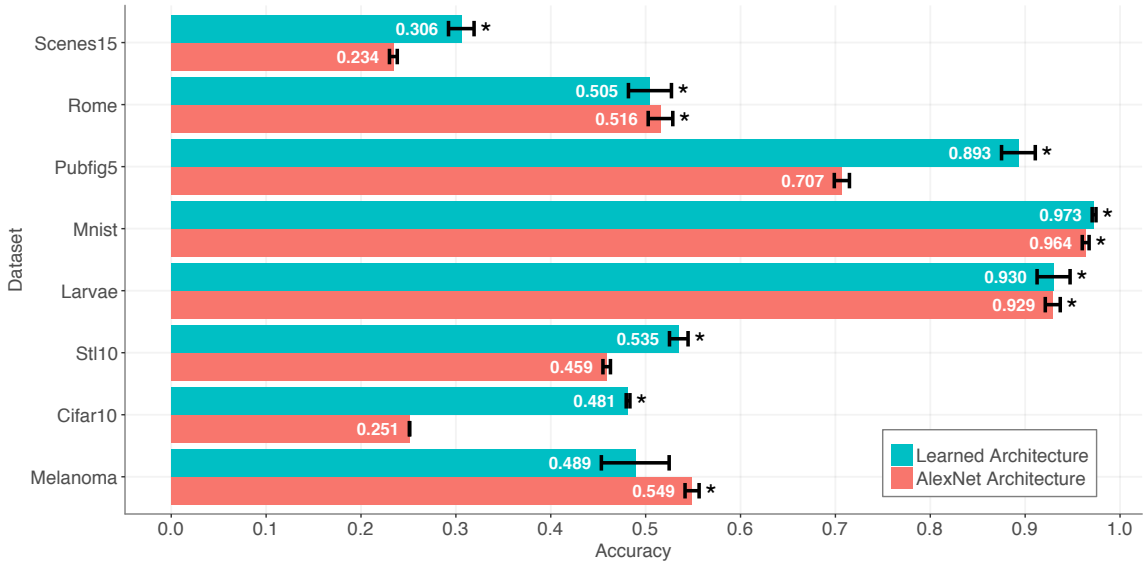


Figure 4.3: Pairwise comparison between AlexNet's architecture and the learned one in accuracy of classification (∗ indicates statistical significance).

for several cases, including Stl10 and Cifar10, whose images have been originally used to design the AlexNet.

Another issue is about the impact of the training set size on AL, FL, and TL. Which are the methods that require large training sets? In order to evaluate this issue, we perform an experiment with Pubfig5, in which the training sets for AL, FL, and TL increase at a rate of 15%, going from 15% to 90% of the data set size, leaving a fixed test set with 10% of the samples. Figure 4.4 presents the respective accuracy curves of AL, FL, and TL.

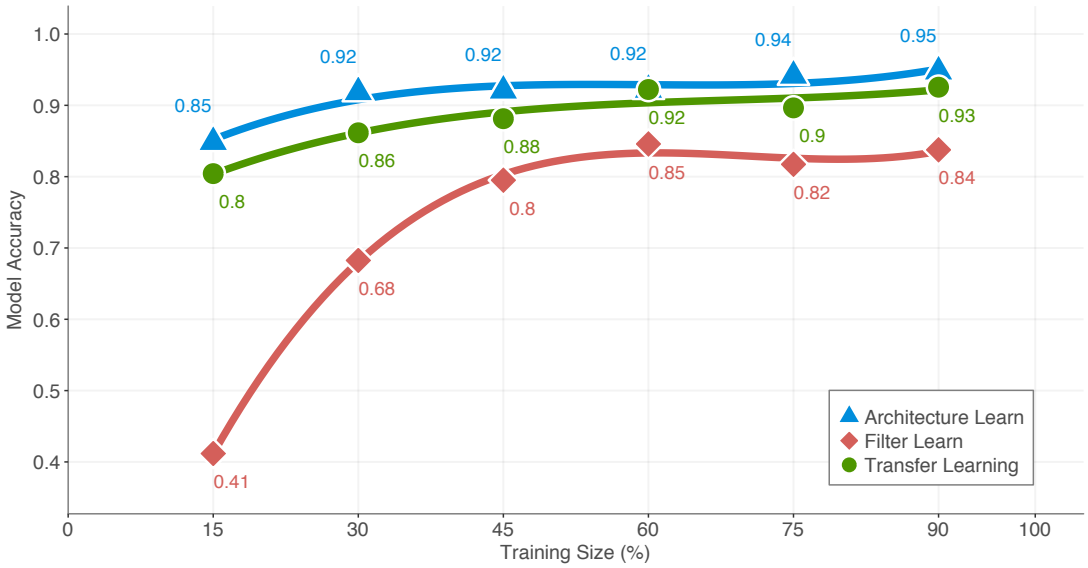


Figure 4.4: Accuracy curves for AL, TL, and FL in Pubfig5, by using an increasing number of training samples.

As we can observe, FL really requires large training sets and the impact of more training samples on TL and AL is considerably less than their impact on FL. The accuracy gain in TL and AL is considerably higher in the beginning of their curves, which suggests they do not need large training sets. In order to confirm this observation for AL, we fix the image descriptor learned with X%, $X = 15, 30, 45, 60, 75, 90$, of the training samples, but project the SVM classifier with more training samples. Table 4.3 shows that above $X = 30$, the accuracy gain is only about 2%. In addition to that, the network architecture selected by AL is consistently the same. However, it does not seem feasible to predict the minimum number of required training samples before their pre-annotation.

Table 4.3: An incremental comparison of sample size effect in architectures learning and the sample size effect in learning the architecture classifier.

Training Size	15%	30%	45%	60%	75%	90%
15%-Descriptor	0.84849	0.80735	0.86716	0.87686	0.87417	0.87684
30%-Descriptor	-	0.91840	0.92050	0.94463	0.94141	0.94730
45%-Descriptor	-	-	0.92050	0.94463	0.94141	0.94730
60%-Descriptor	-	-	-	0.92162	0.92106	0.92485
75%-Descriptor	-	-	-	-	0.94141	0.94730
90%-Descriptor	-	-	-	-	-	0.94730

We may conclude that TL is the best choice under supervised data constraint. However, the effectiveness of the feature learning process can only be confirmed when the number of supervised samples is increased. This issue is addressed next.

4.3 Is the learned feature space effective?

The effectiveness of a feature space may be observed by projecting the training samples and verifying the separability among the categories [72, 73, 74]. However the obtained feature spaces are sparse and highly dimensional, and data projection methods usually depend on distance functions. Another option is to increase the number of supervised training samples used to project the classifier and verify its accuracy gain. Active learning methods are the indicated strategies to reduce human effort in sample annotation. In these methods, at each iteration, an apprentice classifier labels and selects the most informative samples (i.e., those with uncertain labels) for the expert's supervision. The classifier is retrained with the new supervised samples and the process repeats until the user is satisfied.

Considering an SVM classifier, we have evaluated the improvement in classification accuracy when additional informative samples are selected per iteration, based on their distance to the decision boundaries (hyperplanes of each category). We let the

initial training set, used for feature learning, to increase at each iteration by 10 samples per category until it doubles the size. Considering a fixed test set, we evaluate the classification accuracy along iterations, while fixing the previously learned image descriptors from AL, FL, and TL (Figure 4.5[2]). It should be expected an increase in the curve as an indication of the effectiveness of the feature learning approach.

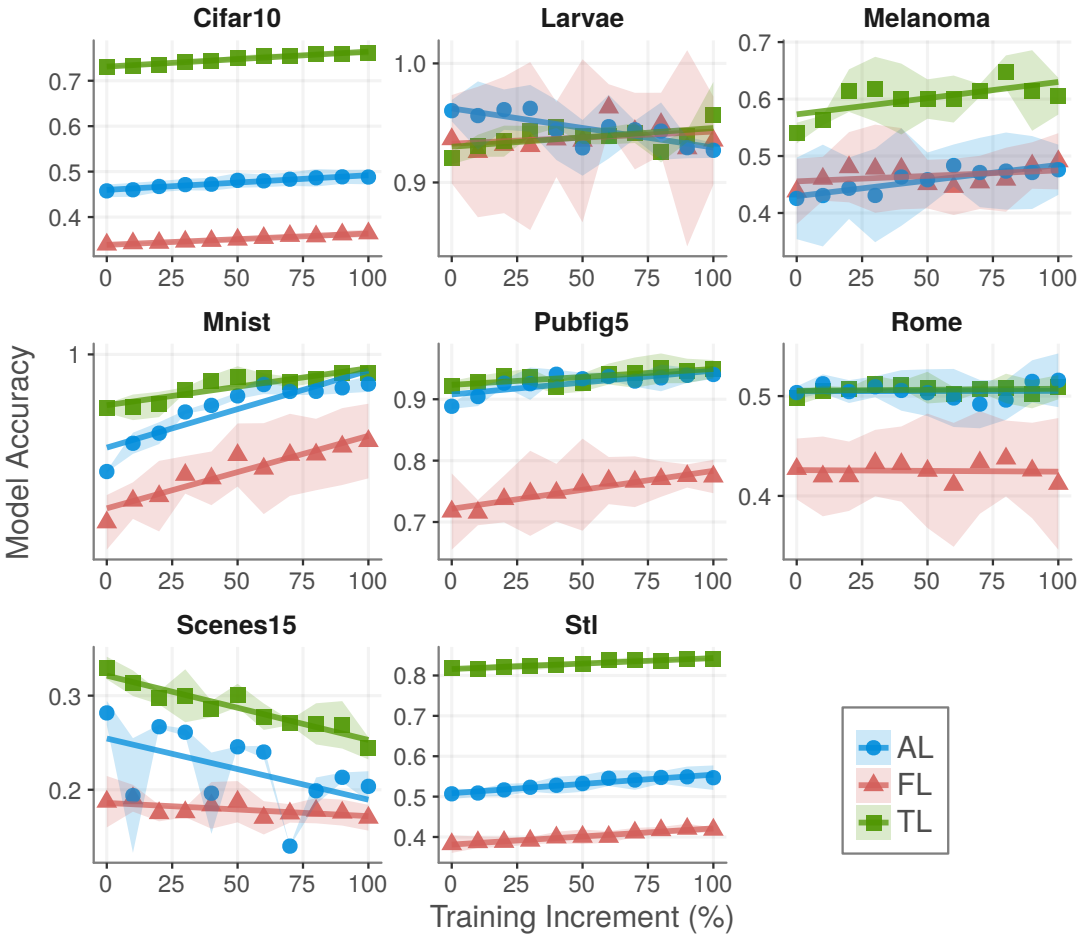


Figure 4.5: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on each data set.

As we can see, for most data sets, all feature learning approaches obtained reasonably effective descriptors (with the exception of Rome and Scenes15). However, it is interesting to observe that, in the case of Larvae and the descriptor from AL, the choice of more informative samples during active learning reduces classification accuracy. In general, TL and AL are the ones that produce the lower standard deviations (as indicated by colored shadow around the curves), which makes them more stable

[2] A better version of this plot is available in Appendix A

descriptors. Note also that the superiority of the descriptor from TL remains during active learning.

It is important to point that, due to the poor performance obtained ofr Rome and Scenes datasets during the feature learning process, both data sets have not shown to benefit from the active learning, when compared with the remaining data sets.

The most important conclusion in this experiment is that we have been able to improve accuracy with the learned descriptors as the number of supervised samples increases. However, we still need to assess if the improvement is possible with artificial data augmentation. This issue is addressed in the next section.

4.4 Does artificial data augmentation improve the results?

Due to the previously mentioned underperformance of Deep Learning under data constraint, a common approach to deal with the problem is to artificially increase the number of samples. Many techniques have been developed to virtually augment data sets, from general approaches, such as affine transformations on images, to expensive 3D models that create different views for face recognition [58, 67].

The benefits of data augmentation on ConvNets have been widely shown in many applications [46, 54, 96], using different approaches to artificially increase the training data set ($\mathcal{Z}_1$). Since it surpasses the scope of this work, the data augmentation approach adopted here relies solely in applying rotation on training images (90°, 180°, 270°), creating data sets 4 times bigger than the data sets adopted in the previous experiments. Increasing the number of training samples, we also increase the time spent in the learning process, while adopting the same learning hyperparameters as before on both, backpropagation and architecture learning. The second reason in experimenting simple data augmentation approaches is to assess if data augmentation approaches will always be able to improve the descriptor learning process.

As ConvNets are usually employed with feature learning and design of the classifier in a same pipeline, it is not usually possible to discriminate the impact of data augmentation on the descriptor or the classifier. In this work, we decouple feature extraction and classification, making it possible to evaluate two different approaches based on data augmentation: (i) Learn the ConvNet features and design the SVM classifier using the augmented data. (ii) Learn the ConvNet features with the augmented data and design the SVM classifier using the original data only. In Figure 4.6 we present a comparison among both approaches and the original one with no data augmentation.

Considering a t-student significance test, FL shows a significant improvement on Rome, Mnist and Larvae data sets, while TL features improves in Rome, Larvae and Melanoma. Data augmentation in AL, on the other hand, does not produce considerable improvement in any data set.

In general, the improvement in classification obtained by the adopted data augmen-

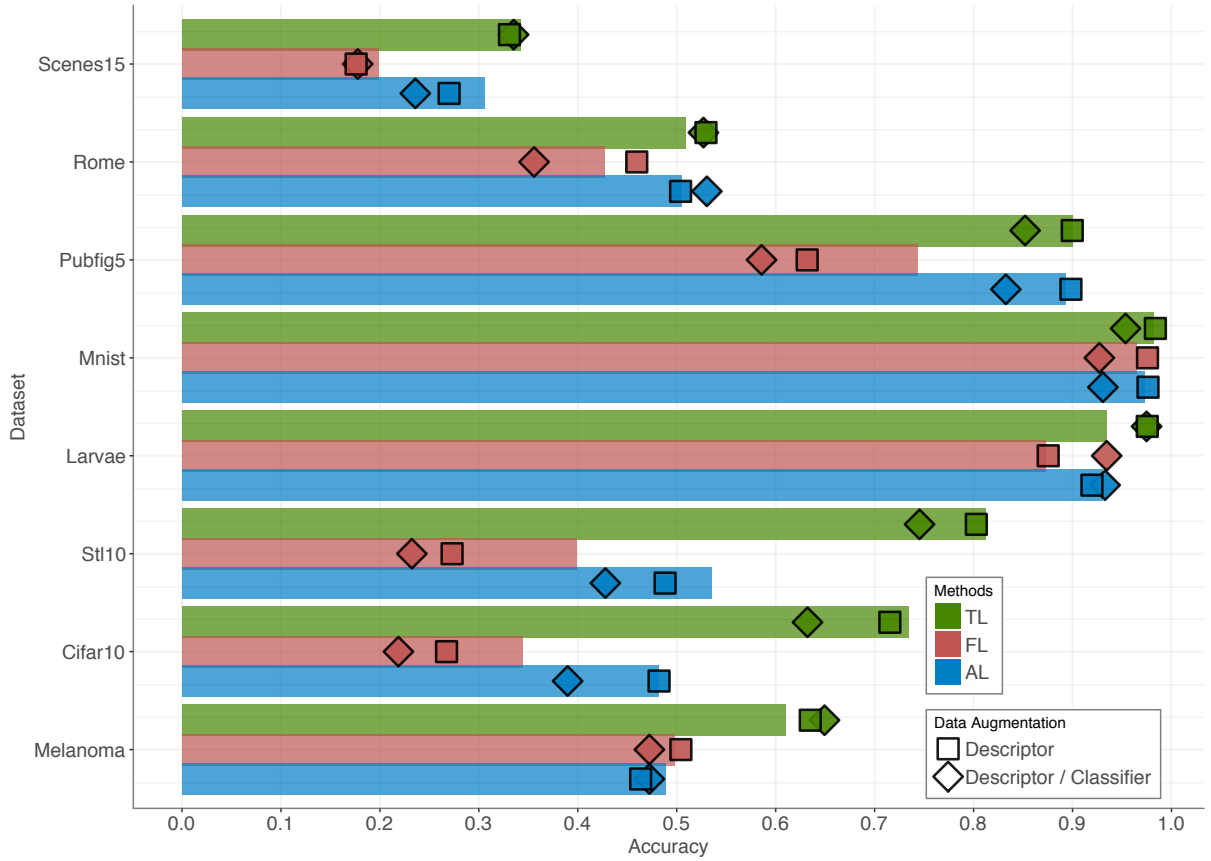


Figure 4.6: Accuracy of Convolution Networks for Filter, Transfer and Architecture Learning, and the accuracy obtained with artificially augmented data for descriptor learning only and for descriptor and classifier training.

tation strategies is marginal. In several cases, they can otherwise considerably reduce performance. One can also notice that, in the cases where data augmentation could benefit the performance, this positive impact is higher in feature learning than when using it for both, feature learning and the classifier training.

ConvNet feature spaces are usually sparse and high dimensional, which impair the performance of distance-based operations, such as clustering and data visualization. This issue is addressed in the next section.

4.5 Can the obtained feature spaces be reduced?

A known problem with the ConvNet approach lies on their high dimensional sparse feature vectors, which strongly limits the use of distance-based classification [2], such as Nearest Neighbors and Optimum-Path Forest [65] classifiers. This also affects the use of clustering algorithms, which are, mostly based in the Euclidean distance for similarity measure [71].

The following experiment is drawn to evaluate the impact of the learned descriptors

on distance-based operators[2], such as the Optimum-Path Forest classifier. As a first scenario, the output of the last convolutional layer is considered as feature vector. We then, evaluate the performance of the Optimum-Path Forest classifier using that feature vector in comparison to the original SVM classification. The results are presented in Figure 4.7.

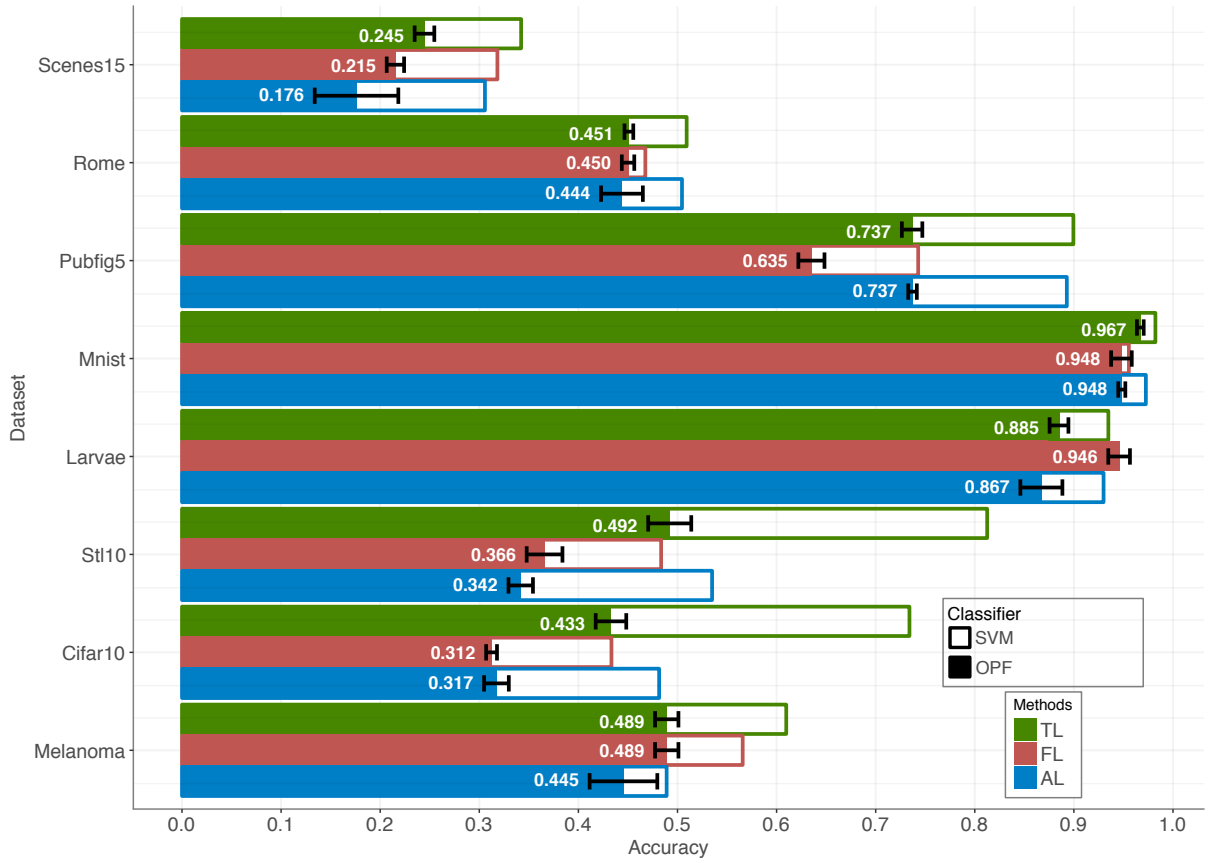


Figure 4.7: Accuracy of Optimum-Path Forest Classifier and Support Vector Machines for all data sets, with the last convolution hidden layer as feature vector.

Except for the Larvae data set, the Transfer Learning approach has shown to find more suitable features for distance-based classification. However, the negative impact of the learned descriptors on OPF is evident.

In order to assess the benefit of the dimensionality reduction on OPF classification, we assess the feature vector that results at the last hidden layer (after feature space reduction by the fully connected layers) of the AlexNet, from TL and FL. Figure 4.8 presents a comparison of OPF classification before and after the fully connected dimensionality reduction.

As we can see, the fully connected dimensionality reduction reduces the negative impact of the ConvNet features on OPF, for both FL and TL features[3]. Cifar10 and

[3]AL is not included in this experiment, since this approach does not use fully connected layers.

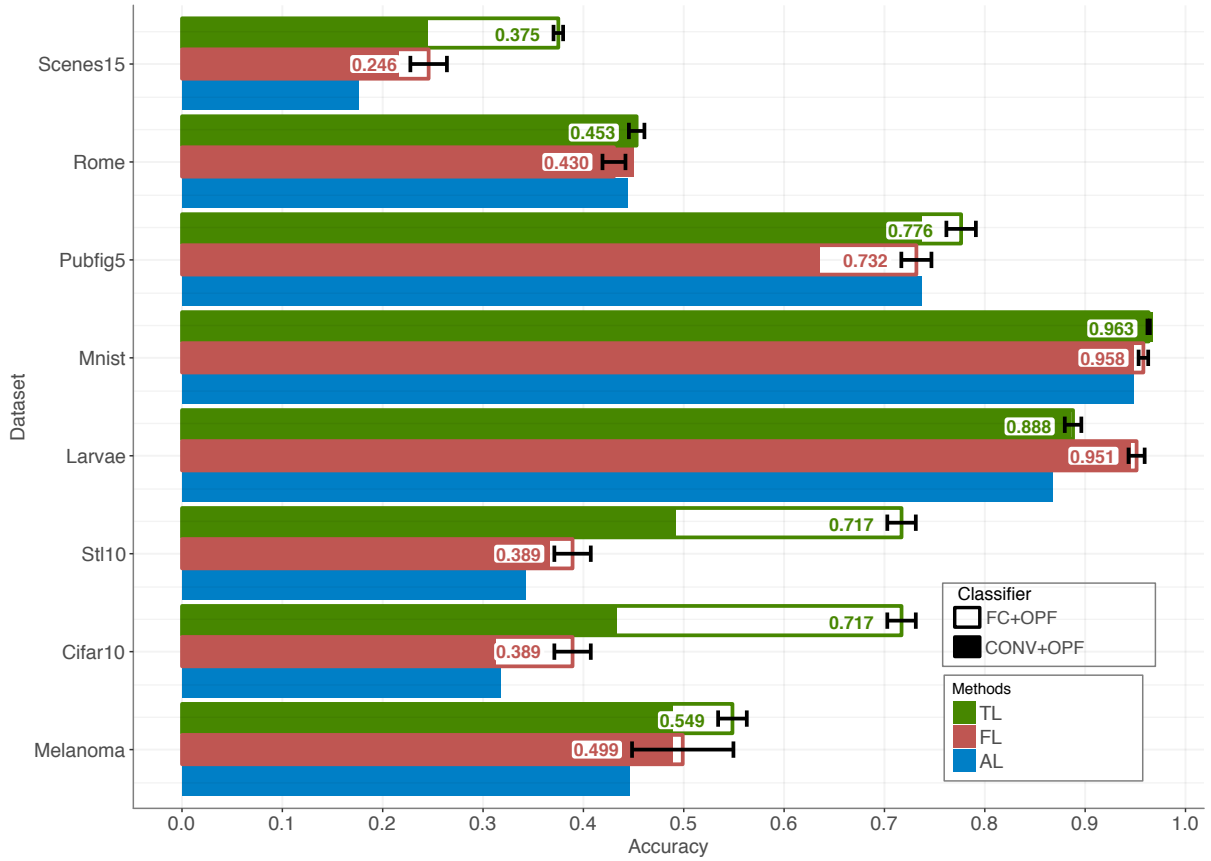


Figure 4.8: The accuracy of Optimum-Path Forest Classifier using the last fully connected hidden layer as feature vector.

Stl10 data sets show better results using the TL features, since AlexNet was originally designed with them.

With the consistent benefit of fully connected dimensionality reduction, one might wonder if conventional dimensionality reduction techniques can also reduce the negative impact on the OPF classification. For this reason, we consider unsupervised and supervised dimensionality reduction approaches, namely, Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) to be applied in the ConvNet descriptor (last convolutional layer). Figure 4.9 presents the accuracy of the OPF classification with such techniques.

For most data sets LDA outperforms PCA, with the exception of Scenes15 and Rome data sets, where, as indicated in the previous experiments, we could not learn a proper descriptor, what considerably impacts the performance of supervised dimensionality reduction techniques.

As we can see, LDA achieves an accuracy rate similar to the fully connected reduction, being able to significantly improve the OPF classification. Since LDA reduces the problem to a $\mathbb{R}^{c-1}$ feature space, for c categories, this is an interesting result in addition to the fact that the fully connected reduction outputs a feature vector

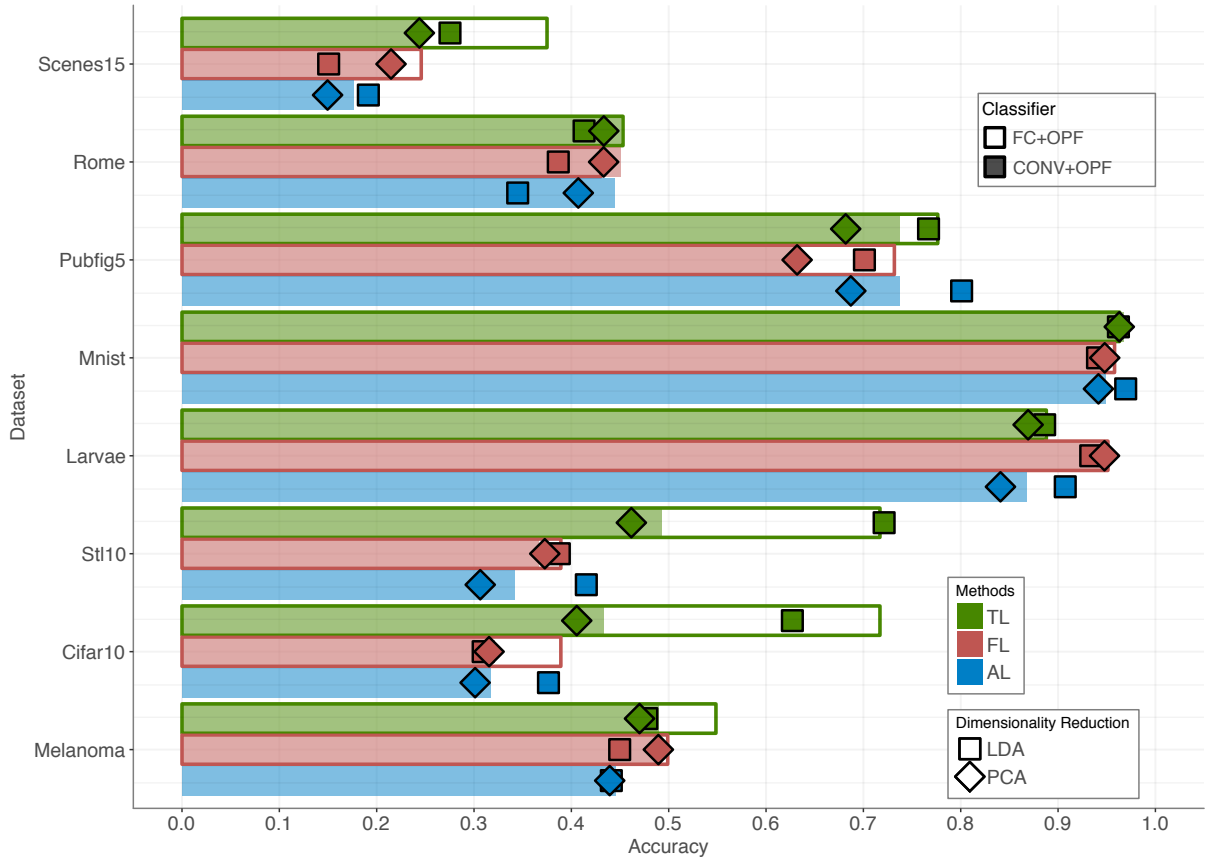


Figure 4.9: The accuracy of Optimum-Path Forest classifier applying PCA and LDA reduction on the output of the last convolutional layer.

in $\mathbb{R}^{1000}$.

Since PCA and LDA can provide considerable feature space reduction, one might wonder if it can perform better if applied after the fully connected reduction. To evaluate this, we have applied PCA and LDA reduction in the output of the last hidden fully connected layer. Figure 4.10 shows the results of this experiment.

The application of PCA/LDA over the feature vector reduced by the fully connected layers does not significant improvement classification performance for most data sets, with exception of the TL descriptor on Cifar10 and Stl10, which, again, represent similar data sets to the AlexNet original training data. Although the performance of the fully connected feature space reduction surpasses LDA, it is important to point out that the feature space reduction in LDA is consideraby more drastic.

Dimensionality reduction also impacts data visualization methods that rely on distance functions to project the high-dimensional feature space on to $\mathbb{R}^2$, allowing humans to understand the structure of the data distribution. The t-Distributed Stochastic Neighbor Embedding (t-SNE) [56] is a recent approach for such data projection, being successfully used to understand the neuron activity of ConvNets during feature learning [74].

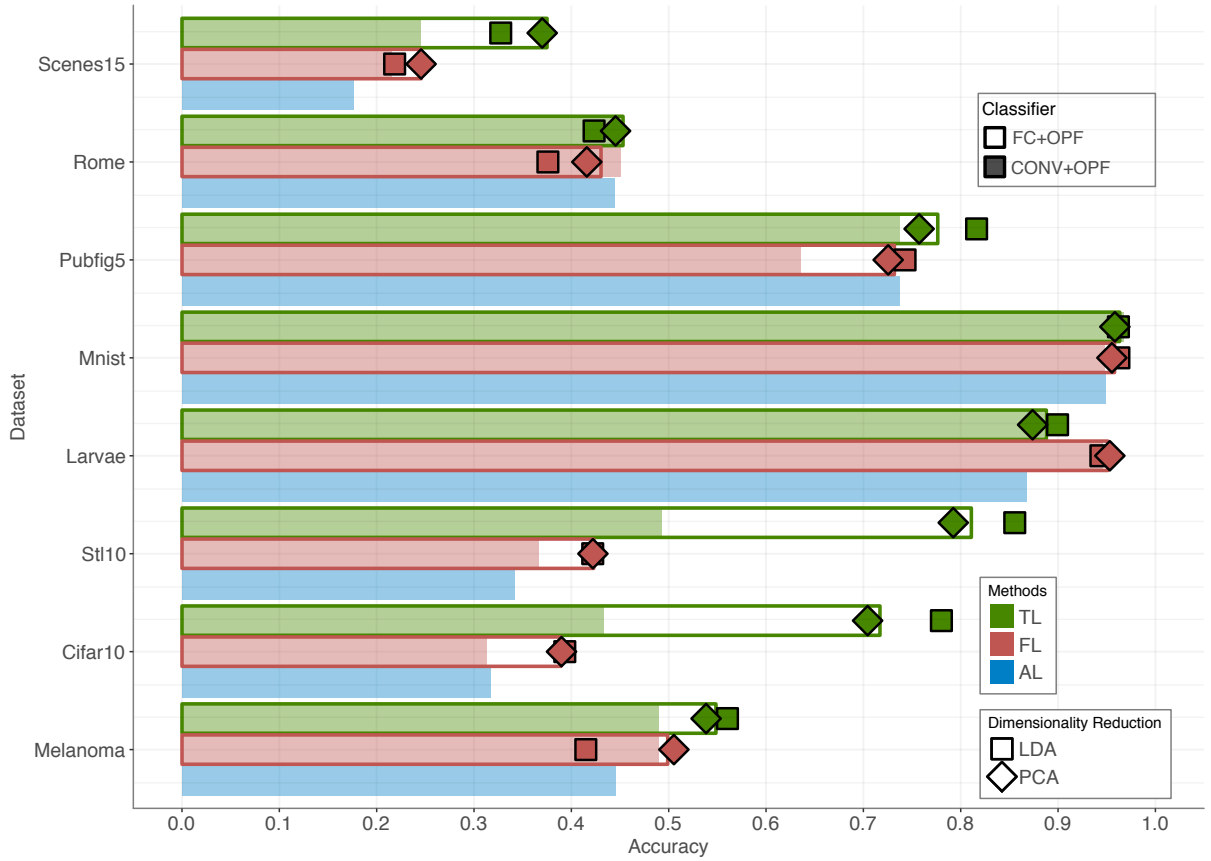


Figure 4.10: The accuracy of Optimum-Path Forest classifier using PCA and LDA to reduce the feature space of the last fully connected hidden layer.

Figure 4.11 [4] presents the t-SNE projection of the high dimensional feature space obtained in the last convolutional layer of the three feature learning approaches, TL, FL, and AL, for the Pubfig5 data set. One can notice that, both TL and AL show a reduced category overlapping when compared with FL. However, while TL is able to create more compact clusters, AL spreads more the projected samples. In all cases, however, the overlapping among the categories is still considerable, indicating a negative impact of the high dimensional feature space on the t-SNE method. In order to alleviate the problem, we repeat the experiment with the resulting feature spaces of the LDA reduction on the descriptor of the last convolutional layer (Figure 4.12[5]).

As we can see, LDA significantly improves the category separation, which explains the accuracy gain obtained with the OPF classifier, indicating that it is possible to find a linear transformation able to properly capture the discrimination power of ConvNet sparse codes, thus opening the application of ConvNets to a wider range of distance-based operators, while holding a reasonably good performance in classification.

[4] A better version of these plots are available in Appendix B
[5] A better version of these plots are available in Appendix B

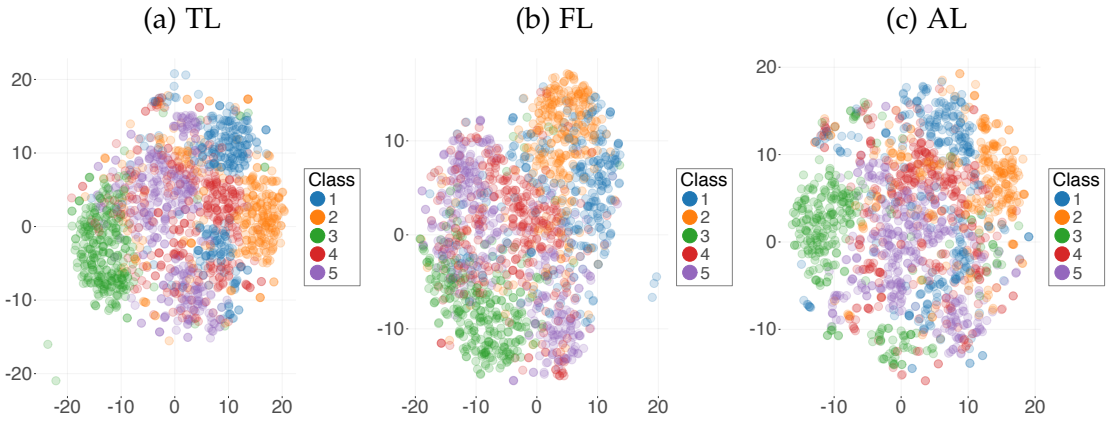


Figure 4.11: t-SNE 2D visualization of all ConvNet descriptors on Pubfig5 dataset.

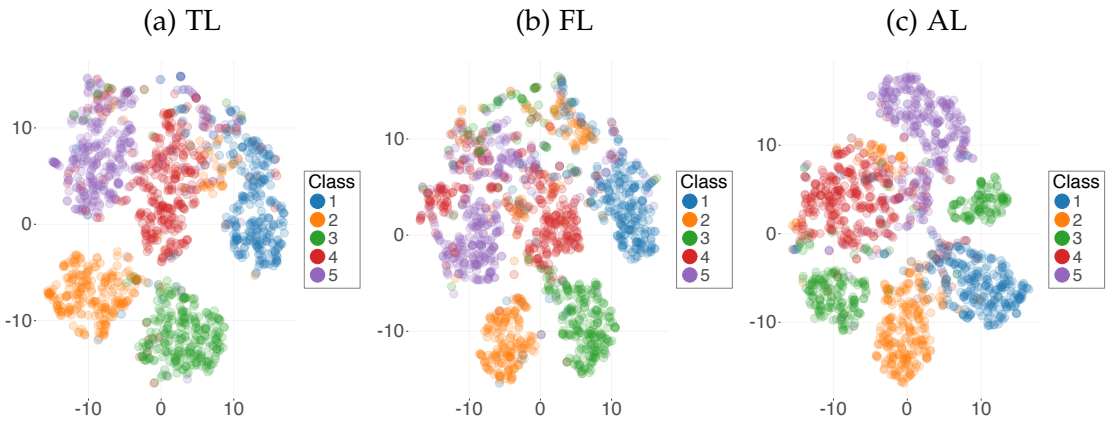


Figure 4.12: t-SNE 2D visualization of the LDA dimensionality reduction of ConvNet descriptors on Pubfig5 dataset.

Chapter 5

Case Study: Diagnosis of Human Intestinal Parasites

"Science is the captain, and practice the soldiers."

—Leonardo da Vinci

This chapter presents an application of the concepts developed so far, applying the Deep Learning concepts in a real scenario for the diagnosis of intestinal parasites. The process of image acquisition is presented, along with the data sets adopted in this experiment. We also compare the learned features against the state of the art knowledge-based features [88] and data-driven features [66], where, even with the data limitation imposed by the problem, ConvNets are able to significantly improve the current model.

5.1 Automatic diagnosis of human intestinal parasites

Infections by human intestinal parasites can cause death, physical and mental disorders in children and immunodeficient adults [70]. Image analysis methods have been proposed [3, 5, 24, 29], in an effort to achieve fast and effective diagnosis. Among the proposed ones, we can highlight the fully automated system developed by Suzuki et al. [88], which is unique in performing image acquisition and analysis with no human intervention.

In a single fecal exam, hundreds of image components (parasites and fecal impurities) can be present per field of the optical microscopy slide. As a first stage, these components must be detected and the parasite candidates must be segmented from the images for subsequent characterization and classification. The image segmentation method, as specified in Suzuki et al. [88], consists of the following steps (illustrated in Figure 5.1):

i. *Quantization*: the colored images are converted to a 64 gray level image, in order to simplify the search for candidate objects.

ii. *Border enhancement*: the borders are enhanced with a Sobel filter.

iii. *Ellipse matching*: the regions with ellipse-like shape are considered parasite candidates, since *protozoa* cysts and *helminth* eggs have circular or elliptical shapes, and *larvae* have elongated regions, where ellipses can fit properly.

iv. *Object delineation*: the matched ellipsis are used to define external and internal markers, allowing a segmentation through the Image Foresting Transform (IFT) algorithm [27].

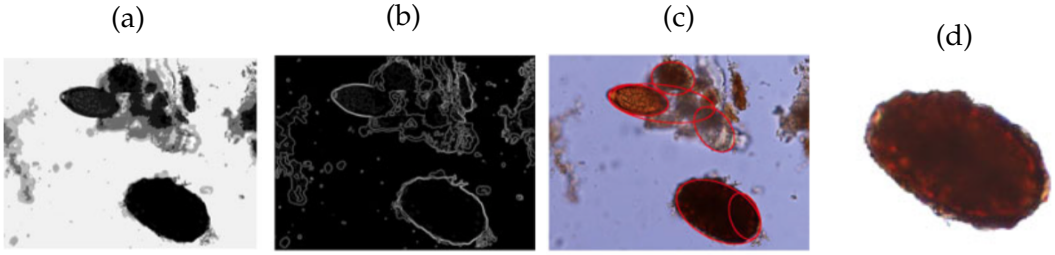


Figure 5.1: (a) Gray level quantization. (b) Sobel border enhancement. (c) Ellipse matching. (d) IFT object delineation. Adapted from [88].

As some species strongly diverge in shape and size, Suzuki et al. [88] divide the parasite species in three different groups, for better categorization:

i. Larvae: composed by *S. stercoralis* and impurities of similar size and shape,

ii. Helminth eggs: composed by *H. nana*, *H. diminuta*, *Ancilostomideo*, *E. vermicularis*, *A. lumbricoides*, *T. trichiura*, *S. mansoni*, *Taenia* and impurities of similar size and shape,

iii. Protozoa: composed by *E. coli*, *E. histolytica*, *E.nana*, *Giardia*, *I.butschlii*, *B.hominis* and impurities of similar size and shape.

All experiments in this chapter are performed in a data set composed of 16.437 candidate objects of fecal impurities and the 15 most common species of intestinal parasites in Brazil. This is the same data set considered in [88], and some sample examples are presented in Figure 5.2.

An important aspect of this data set, lies on its strong category imbalance, caused by an uneven *a priori* probability of infection by different species, as well as an exceedingly number of impurities against true parasites. Table 5.1 presents the distribution of samples per category in the given data set.

Figure 5.2: (a) Examples of some of the most common parasite species. (b) Impurities with size and shape similar to parasites.

Table 5.1: Distribution of available samples in the parasites data set, according to parasite species and group.

Protozoa		**Helminth eggs**		**Larvae**	
E. coli	719	*H. nana*	348	*S. stercoralis*	446
E. histolytica	78	*H. diminuta*	80	Impurities	3068
E.nana	724	*Ancilostomideo*	148		
Giardia	641	*E. vermicularis*	122		
I.butschlii	1501	*A. lumbricoides*	337		
B.hominis	189	*T. trichiura*	375		
Impurities	5716	*S. mansoni*	122		
		Taenia	236		
		Impurities	3344		
Total	**9568**	**Total**	**5112**	**Total**	**3544**

As we can notice, for some species, less than 100 samples are available for training, evaluation and test, due to a naturally lower occurrence of such species. As mentioned before, the category balance poses an important requirement for a good performance of ConvNets. The experiments on ConvNets and the analysis of their results are presented in the following section.

5.2 Classification of intestinal parasites based on ConvNets

In order to improve the automated diagnosis system of intestinal parasites, as proposed in Suzuki et al. [88], Peixinho et al. [66] presented a ConvNet descriptor based on Architecture Learning (AL). This descriptor is able to reasonably improve classification

performance, when the classifier is trained with a sufficient number of samples obtained by artificial data augmentation. Due to the strong category imbalance previously mentioned in Section 5.1, this early Deep Learning solution learns the ConvNet architecture by considering balanced subsets for each category with 5% of the samples for training and 5% of the samples for evaluation (a total of 10% of the samples are used to learn the descriptor). However, in order to achieve a reasonable classification accuracy, the number of samples used to train the classifier is increased to 50% of the samples, as in the state-of-the-art approach [88].

In this chapter we add the Transfer Learning (TL) of AlexNet, which has shown, in the previous experiments, to perform better in scenarios where the available data is strongly restricted. Considering the natural restriction imposed by the category imbalance, the ConvNet descriptors are learned based on TL and AL with a small number of training samples. For instance, we set a number of 70 samples per category for the Helminth and Protozoa groups, and 100 samples for the Larvae group, since this last group has more available samples.

In the following experiments we compare TL with the knowledge-based and data-driven (AL) descriptors [66, 88]. Although the original approach, developed by Suzuki et al. [88], relies on the Optimum-Path Forest [65] classifier, we are using the SVM classifier due to the ConvNet-based features, thus providing a fair comparison among the descriptors. Table 5.2 presents the classification accuracy achieved for all descriptors.

Table 5.2: Classification **accuracies** using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **under supervised data constraint**. (Bold values indicates statistical significance).

Dataset	Transfer Learn	Arch Learn [66]	Suzuki et al. [88]
Larvae	$\mathbf{0.974 \pm 0.002}$	0.931 ± 0.009	0.936 ± 0.020
Protozoa	$\mathbf{0.959 \pm 0.004}$	0.915 ± 0.004	0.903 ± 0.007
Helmints	$\mathbf{0.978 \pm 0.007}$	0.932 ± 0.005	0.931 ± 0.006

The AL and the handcrafted descriptors achieve similar performance in the scenario under supervised data constraint, while TL is able to significantly outperform them in all groups. However, dealing with medical diagnosis, it is important to evaluate how this outperformance reflects in precision and recall for each specie. We present the comparative results using precision and recall for each specie in Tables 5.3 and 5.4, respectively.

Although the recall has shown similar results for all descriptors, being the TL descriptor able to outperform the others for only 3 out of 15 categories, the TL precision is better for most species, by significantly reducing the number of false positives.

In order to properly show the precision and recall improvement obtained with the Deep Learning approaches in a scalar measure, we also evaluate the descriptors with

Table 5.3: Classification **precision** using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **under supervised data constraint**. (Bold values indicates statistical significance).

Species	Transfer Learn.	Arch Learn [66]	Suzuki et al. [88]
E. coli	$\mathbf{0.860 \pm 0.034}$	0.795 ± 0.037	0.731 ± 0.066
E. histolytica	$\mathbf{0.083 \pm 0.020}$	0.057 ± 0.010	0.047 ± 0.013
E.nana	0.785 ± 0.034	0.658 ± 0.048	0.636 ± 0.074
Giardia	$\mathbf{0.742 \pm 0.038}$	0.570 ± 0.039	0.516 ± 0.053
I.butschlii	$\mathbf{0.927 \pm 0.024}$	0.747 ± 0.042	0.746 ± 0.040
B.hominis	0.363 ± 0.047	0.354 ± 0.066	0.274 ± 0.028
Protozoa Impurities	0.997 ± 0.001	0.984 ± 0.004	0.986 ± 0.004
H. nana	$\mathbf{0.942 \pm 0.032}$	0.737 ± 0.065	0.694 ± 0.113
H. diminuta	$\mathbf{0.353 \pm 0.092}$	0.127 ± 0.025	0.132 ± 0.023
Ancilostomideo	0.554 ± 0.104	0.333 ± 0.086	0.440 ± 0.038
E. vermicularis	$\mathbf{0.879 \pm 0.061}$	0.266 ± 0.040	0.323 ± 0.042
A. lumbricoides	0.605 ± 0.099	0.524 ± 0.084	0.504 ± 0.079
T. trichiura	$\mathbf{0.913 \pm 0.030}$	0.809 ± 0.029	0.680 ± 0.087
S. mansoni	$\mathbf{0.414 \pm 0.061}$	0.267 ± 0.046	0.136 ± 0.049
Taenia	$\mathbf{0.797 \pm 0.049}$	0.645 ± 0.070	0.499 ± 0.093
Helminth Impurities	0.998 ± 0.001	0.991 ± 0.002	0.993 ± 0.005
S. stercorali	$\mathbf{0.762 \pm 0.047}$	0.475 ± 0.050	0.527 ± 0.116
Larvae Impurities	0.998 ± 0.001	0.999 ± 0.001	0.998 ± 0.002

the Cohen's Kappa statistic κ, which is known to be strongly related to the ROC curve [9], being a more suitable metric. In Table 5.6, we present the Kappa statistic for each descriptor, reinforcing the superior performance of TL.

In a scenario with supervised data restriction, the TL descriptor can significantly improve classification performance, surpassing both, the handcrafted descriptor [88], and the AL descriptor [66]. However, Suzuki et al. [88] evaluates its descriptor in a scenario with more available data, since, as a knowledge-based descriptor, it does not suffer from the strong category imbalance present on the parasite data set.

By increasing the number of training samples, we now compare the performances of the descriptors similarly to the scenario proposed in Suzuki et al.[88]. Each group is divided into 50% of training samples and 50% of test samples by random sampling, with no category balance imposition. We keep the descriptors as previously learned, while the classifier is retrained with more samples. Tables 5.7 and 5.8 present the results in accuracy and kappa statistic.

Table 5.4: Classification **recall** using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **under supervised data constraint**. (Bold values indicates statistical significance).

Species	Transfer Learn.	Arch Learn. [66]	Suzuki et al. [88]
E. coli	0.995 ± 0.002	0.986 ± 0.010	0.984 ± 0.005
E. histolytica	1.000 ± 0.000	1.000 ± 0.000	0.975 ± 0.050
E.nana	0.971 ± 0.010	0.907 ± 0.031	0.932 ± 0.028
Giardia	0.965 ± 0.009	0.955 ± 0.010	0.926 ± 0.017
I.butschlii	0.946 ± 0.010	0.895 ± 0.020	0.875 ± 0.019
B.hominis	0.973 ± 0.012	0.928 ± 0.033	0.948 ± 0.015
Protozoa Impurities	0.862 ± 0.029	0.735 ± 0.023	0.682 ± 0.030
H. nana	0.994 ± 0.005	0.986 ± 0.006	0.963 ± 0.027
H. diminuta	0.980 ± 0.040	0.980 ± 0.040	0.960 ± 0.049
Ancilostomideo	0.990 ± 0.010	0.928 ± 0.048	0.977 ± 0.013
E. vermicularis	0.996 ± 0.008	0.977 ± 0.019	0.981 ± 0.017
A. lumbricoides	0.983 ± 0.010	0.894 ± 0.030	0.958 ± 0.022
T. trichiura	$\mathbf{0.997 \pm 0.002}$	0.988 ± 0.006	0.974 ± 0.008
S. mansoni	1.000 ± 0.000	0.950 ± 0.029	0.985 ± 0.014
Taenia	0.994 ± 0.000	0.989 ± 0.005	0.993 ± 0.005
Helminth Impurities	$\mathbf{0.865 \pm 0.032}$	0.699 ± 0.027	0.592 ± 0.087
S. stercorali	0.985 ± 0.009	0.992 ± 0.006	0.987 ± 0.012
Larvae Impurities	$\mathbf{0.963 \pm 0.010}$	0.869 ± 0.024	0.886 ± 0.047

Table 5.5: The Cohen's ***Kappa*** statistic of classification using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **under supervised data constraint**.

Table 5.6: A comparison of between the ConvNet and the state-of-the-art descriptor for automatic diagnosis [88]. (Bold values indicates statistical significance).

Dataset	TL	AL [66]	Suzuki et al. [88]
Larvae	$\mathbf{0.839 \pm 0.031}$	0.581 ± 0.052	0.626 ± 0.113
Protozoa	$\mathbf{0.841 \pm 0.026}$	0.708 ± 0.015	0.664 ± 0.021
Helmints	$\mathbf{0.808 \pm 0.038}$	0.618 ± 0.023	0.539 ± 0.061

With more training samples, a significant improvement in accuracy is achieved for AL and the handcrafted descriptors. The classification accuracy with the TL descriptor, however, has not improved much. The differences among the methods is better perceived when using the kappa score. Tables 5.9 and 5.10 also show similar

Table 5.7: Classification **accuracy** using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **with no data constraint**. (Bold values indicates statistical significance).

Dataset	TL	AL [66]	Suzuki et al. [88]
Larvae	$\mathbf{0.976 \pm 0.005}$	0.948 ± 0.014	0.921 ± 0.012
Protozoa	$\mathbf{0.961 \pm 0.006}$	0.940 ± 0.006	0.914 ± 0.008
Helmint eggs	$\mathbf{0.977 \pm 0.005}$	0.933 ± 0.006	0.900 ± 0.004

Table 5.8: The Cohen's ***Kappa*** statistic of classification using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **with no data constraint**. (Bold values indicates statistical significance).

Dataset	TL	AL [66]	Suzuki et al. [88]
Larvae	$\mathbf{0.974 \pm 0.002}$	0.878 ± 0.026	0.898 ± 0.017
Protozoa	$\mathbf{0.974 \pm 0.002}$	0.958 ± 0.003	0.919 ± 0.002
Helminth eggs	$\mathbf{0.978 \pm 0.004}$	0.941 ± 0.006	0.918 ± 0.003

results for precision and recall, respectively, when using more training samples.

The results indicate that, based on the Cohen's kappa, the precision and the recall scores, TL is the best candidate to improve the automated diagnosis system of human intestinal parasites.

Now, in order to understand the presented results by data visualization, we analyze the data projection of the handcrafted and ConvNet feature spaces in the next section.

Table 5.9: Classification **precision** using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **with no data constraint**. (Bold values indicates statistical significance).

Species	TL.	AL [66]	Suzuki et al. [88]
E. coli	0.987 ± 0.005	0.977 ± 0.009	0.963 ± 0.009
E. histolytica	0.975 ± 0.022	0.968 ± 0.019	0.863 ± 0.082
E.nana	$\mathbf{0.965 \pm 0.005}$	0.949 ± 0.009	0.904 ± 0.014
Giardia	0.966 ± 0.008	0.958 ± 0.011	0.930 ± 0.020
I.butschlii	0.982 ± 0.004	0.975 ± 0.004	0.921 ± 0.011
B.hominis	0.960 ± 0.015	0.947 ± 0.031	0.895 ± 0.034
Protozoa Impurities	$\mathbf{0.990 \pm 0.001}$	0.981 ± 0.002	0.969 ± 0.001
H. nana	0.986 ± 0.006	0.984 ± 0.016	0.958 ± 0.012
H. diminuta	$\mathbf{1.000 \pm 0.000}$	0.910 ± 0.030	0.939 ± 0.035
Ancilostomideo	0.954 ± 0.017	0.960 ± 0.033	0.946 ± 0.033
E. vermicularis	$\mathbf{1.000 \pm 0.000}$	0.966 ± 0.011	0.936 ± 0.014
A. lumbricoides	$\mathbf{0.976 \pm 0.014}$	0.938 ± 0.017	0.940 ± 0.015
T. trichiura	0.994 ± 0.005	0.988 ± 0.002	0.977 ± 0.007
S. mansoni	$\mathbf{0.986 \pm 0.013}$	0.920 ± 0.029	0.883 ± 0.032
Taenia	0.972 ± 0.013	0.952 ± 0.007	0.972 ± 0.015
Helminth Impurities	$\mathbf{0.990 \pm 0.002}$	0.971 ± 0.003	0.957 ± 0.003
S. stercorali	0.943 ± 0.022	0.881 ± 0.068	0.989 ± 0.008
Larvae Impurities	0.994 ± 0.002	0.988 ± 0.006	0.978 ± 0.003

Table 5.10: Classification **recall** using the ConvNet-based descriptors and the handcrafted descriptor proposed in [88], **with no data constraint**. (Bold values indicates statistical significance).

Species	Transfer Learn.	Arch Learn. [66]	Suzuki et al. [88]
E. coli	$\mathbf{0.996 \pm 0.003}$	0.989 ± 0.004	0.971 ± 0.004
E. histolytica	0.954 ± 0.019	0.908 ± 0.045	0.887 ± 0.058
E.nana	$\mathbf{0.975 \pm 0.008}$	0.944 ± 0.017	0.923 ± 0.019
Giardia	$\mathbf{0.969 \pm 0.003}$	0.959 ± 0.006	0.912 ± 0.015
I.butschlii	0.983 ± 0.003	0.977 ± 0.005	0.928 ± 0.013
B.hominis	0.862 ± 0.039	0.839 ± 0.034	0.812 ± 0.032
Protozoa Impurities	$\mathbf{0.990 \pm 0.001}$	0.985 ± 0.003	0.968 ± 0.001
H. nana	0.991 ± 0.006	0.979 ± 0.011	0.952 ± 0.016
H. diminuta	0.980 ± 0.019	0.960 ± 0.020	0.905 ± 0.043
Ancilostomideo	$\mathbf{0.946 \pm 0.015}$	0.846 ± 0.044	0.862 ± 0.048
E. vermicularis	0.997 ± 0.007	0.934 ± 0.057	0.921 ± 0.041
A. lumbricoides	$\mathbf{0.962 \pm 0.012}$	0.877 ± 0.023	0.894 ± 0.013
T. trichiura	0.999 ± 0.002	0.994 ± 0.006	0.965 ± 0.010
S. mansoni	$\mathbf{0.928 \pm 0.034}$	0.852 ± 0.018	0.695 ± 0.088
Taenia	$\mathbf{0.998 \pm 0.003}$	0.968 ± 0.012	0.925 ± 0.011
Helminth Impurities	$\mathbf{0.992 \pm 0.002}$	0.984 ± 0.005	0.979 ± 0.004
S. stercorali	0.960 ± 0.012	0.915 ± 0.042	0.843 ± 0.023
Larvae Impurities	0.992 ± 0.003	0.980 ± 0.015	0.999 ± 0.001

5.3 Understanding the results by visual analytics

In [72, 73], it was demonstrated that a good separation among classes in the 2D sample projection by t-SNE of a high-dimensional feature space is strongly related to a good classification performance in the original feature space. At the same time, a bad separation among classes in the 2D sample projection is meaningless. In this section, we want to verify if the performance of the image descriptors in the previous section is related to the separation among classes in their corresponding high-dimensional feature spaces.

First, we use the t-SNE[56] to project in $\mathbb{R}^2$ the samples of each group in the feature space of the knowledge-based image descriptor (Figure 5.3). The fecal impurities, in gray, represent a diverse class with examples that can be similar to the ones of any other category. Nevertheless, it is possible to understand that most samples from distinct classes are separated in the high-dimensional feature space. This becomes clearer when we overlook the impurities, as shown in Figure 5.4.

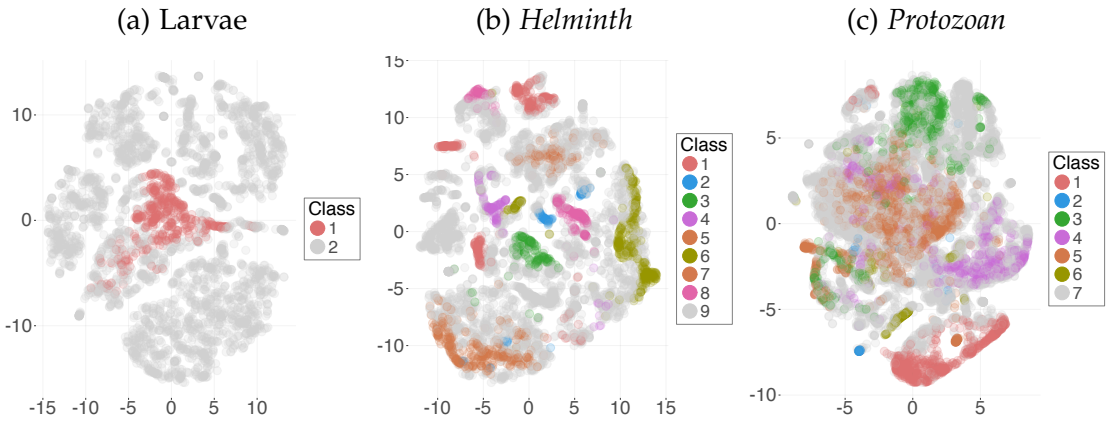


Figure 5.3: The 2D projection of the samples in the feature space of the knowledge-based image descriptor [88], for each group of parasites.

Given that the dimension of the TL feature space is considerably higher than the one of the knowledge-based image descriptor, we apply LDA to reduce that feature space before final projection by t-SNE. The LDA feature space reduction is learned over 50% of the training images only. Figures 5.5 and 5.6 show the 2D sample projections of the TL feature space, for each group, with and without fecal impurities, respectively.

We can observe that the intermediate feature space reduction based on LDA allows to reveal a considerably better separation among the classes in the TL feature space than in the feature space of the knowledge-based image descriptor, which is consistent to the classification results presented in the previous section.

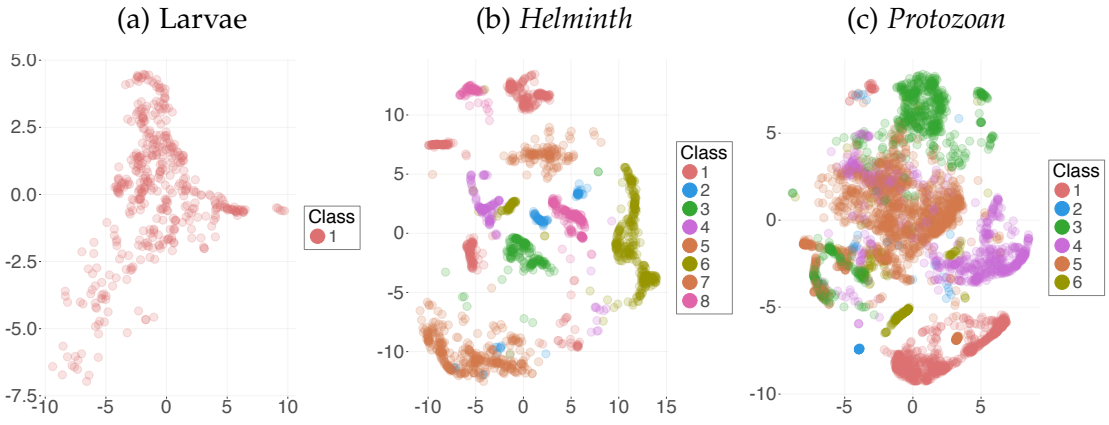


Figure 5.4: The 2D projection of the samples in the feature space of the knowledge-based image descriptor [88], for each group of parasites, but **overlooking the fecal impurities**.

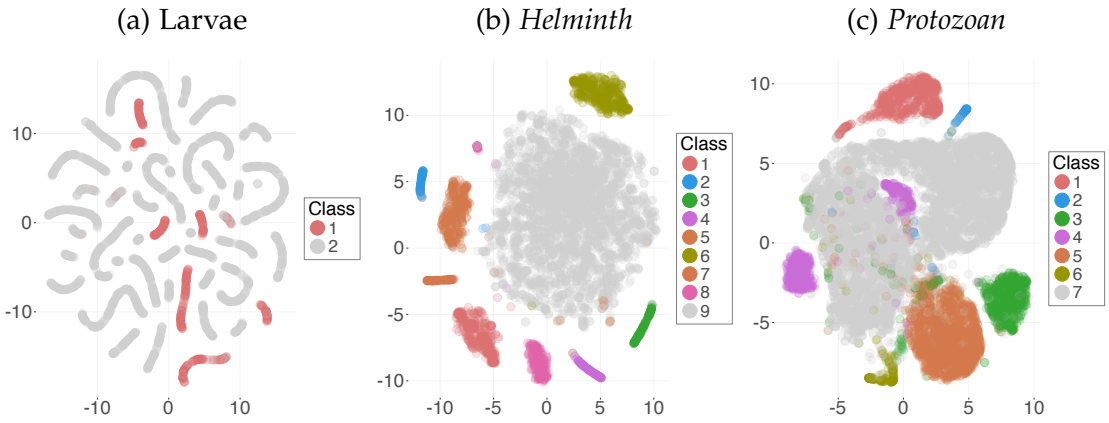


Figure 5.5: The LDA followed by the 2D t-SNE projection of the samples in the feature space of the TL image descriptor, for each group of parasites.

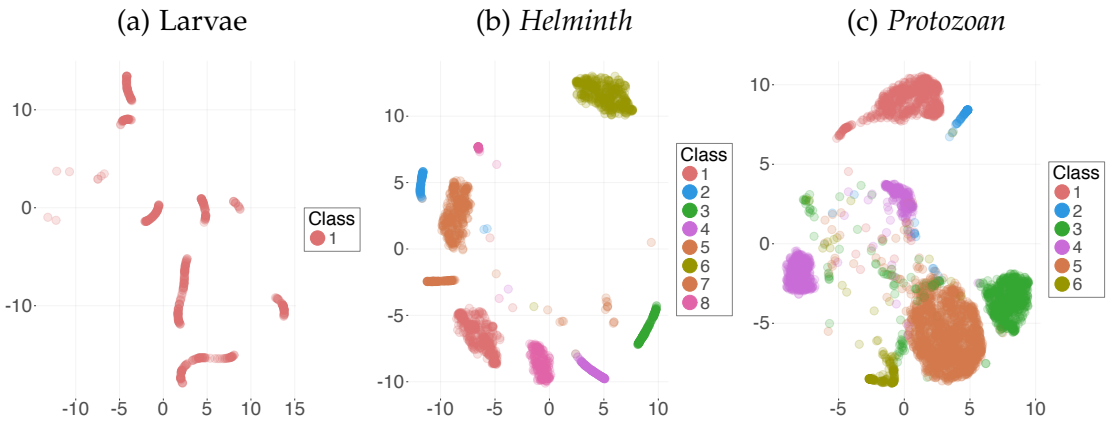


Figure 5.6: The LDA followed by the 2D t-SNE projection of the samples in the feature space of the TL image descriptor, for each group of parasites, but **overlooking the fecal impurities**.

Chapter 6

Conclusion

"We can only see a short distance ahead, but we can see plenty there that needs to be done."

—Alan Turing

Deep feature learning techniques can address pattern recognition problems from many areas of Sciences and Engineering, in which decision-making and decision-support systems based on image analysis rely on an effective feature space representation of the image content for pattern classification. However, the need for large training sets with pre-annotated images is a well known problem.

In this work, we studied three possible solutions using Convolution Networks (ConvNets) under a limited number of supervised images: Transfer Learning (TL), Architecture Learning (AL), and Filter Learning (FL). We have also evaluated the impact of increasing the training set size with no user supervision by artificial data augmentation. The data augmentation strategies based on image rotation are not in general justified to improve performance of the system. We have shown that TL from the AlexNet can be effectively applied to learn features from image data sets with different characteristics, being the best approach in comparison with AL and FL. At the same time, in the absence of a previously learned network for TL, AL is the best alternative under supervised data constraint in comparison with FL.

The performance order of the feature learning techniques was also visually verified by the separation among classes in the 2D projections of their corresponding feature spaces, as proposed in [72, 73, 74]. This suggests expert's intervention for data augmentation as future work. The expert can delineate the regions where unsupervised samples are more likely to come from each class. We believe that data augmentation based on the actual unsupervised images will improve deep feature learning, especially when the accuracy of classification without data augmentation is low.

In order to use distance-based techniques for data visualization, as well as for pattern classification, the reduction of the high-dimensional feature space created by ConvNets is an important issue. The space reduction provided by subsequent fully connected layers and by Linear Discriminant Analysis (LDA) are equally suitable to

address the problem. However, the considerable dimensionality reduction of LDA makes data visualization more efficient. In addition to the efficiency gain, the accuracy of the SVM classification using the high-dimensional feature space was slightly affected by feature space reduction followed by OPF classification. This positive result suggests the further investigation of feature space reduction techniques and their application to other distance-based operations, such as the data organization by clustering, with possible data reduction, for more effective and efficient active learning [80, 81]. By using active learning, the expert can select more relevant samples for supervision and/or data augmentation, and the feature learning process can be revisited for feature improvement. Indeed, we have demonstrated that active learning can improve the classification accuracy of SVM as those relevant samples are selected for label supervision. The interplaying between feature learning and active learning strategies is an interesting research topic, which can reduce human effort and time in data annotation in order to design effective decision-making (-support) systems.

Finally, we have validated TL and AL in a real application — the diagnosis of the 15 most common species of intestinal parasites, distinguishing them from the numerous and diverse class of fecal impurities. TL presented the best result among the compared methods from the state-of-the-art, with and without supervised data constraint. This demonstrates the potential of TL using ConvNets for this application.

In addition to the above directions to future research, there are many other possibilities to improve this work. Unsupervised learning techniques, such as Visual Dictionaries (BoVWs), may be useful to start the process with active learning in order to select the most relevant samples for deep feature learning. The use of data visualization in both, active learning and feature learning, may be exploited to let the expert intervene in both by selecting unsupervised samples and by changing the parameters/hyperparameters of the system. In the end, we expect to build decision-making and -support systems in which the experts can understand the advantages and limitations of the machine, and consequently know when they can rely on the machine's actions.

Bibliography

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2274–2282, 2012.

[2] C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*, pages 420–434. Springer, 2001.

[3] R. S. H. Al-Sameraai, G. K. Hawari, and M. Zeehaida. Automated system for diagnosis intestinal parasites by computerized image analysis. *Modern Applied Science*, 7(5):98–114, 2013.

[4] S. Arora. *Minimizing the costs in generalized interactive annotation learning*. PhD thesis, Carnegie Mellon University, 2012.

[5] D. Avci and A. Varol. An expert diagnosis system for classification of human parasite eggs based on multi-class svm. *Expert Systems with Applications*, 36(1): 43–48, 2009.

[6] S. Avila, N. Thome, M. Cord, E. Valle, and A. D. A. AraúJo. Pooling in image representation: The visual codeword point of view. *Computer Vision and Image Understanding*, 117(5):453–465, 2013.

[7] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. *Unsupervised and Transfer Learning Challenges in Machine Learning, Volume 7*, page 43, 2012.

[8] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[9] A. Ben-David. About the relationship between {ROC} curves and cohen's kappa. *Engineering Applications of Artificial Intelligence*, 21(6):874 – 882, 2008. ISSN 0952-1976.

[10] Y. Bengio and Y. Lecun. Convolutional networks for images, speech, and time-series, 1995.

[11] Y. Bengio, Y. LeCun, et al. Scaling learning algorithms towards ai. *Large-scale kernel machines*, 34(5), 2007.

[12] J. Bergstra, D. Yamins, and D. D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. *ICML (1)*, 28:115–123, 2013.

[13] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

[14] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2559–2566. IEEE, 2010.

[15] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4-5):291–294, 1988.

[16] A. Brew, D. Greene, and P. Cunningham. The interaction between supervised learning and crowdsourcing. In *NIPS workshop on computational social science and the wisdom of crowds*, 2010.

[17] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki. Scene labeling with lstm recurrent neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3547–3555, 2015.

[18] G. Chiachia, A. X. Falcão, N. Pinto, A. Rocha, and D. Cox. Learning person-specific representations from faces in the wild. *IEEE Transactions on Information Forensics and Security (TIFs)*, 9(12):2089–2099, Dec 2014a.

[19] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[20] A. Coates, A. Y. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics*, pages 215–223, 2011.

[21] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, Sept. 1995. ISSN 0885-6125.

[22] D. Cox and N. Pinto. Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 8–15. IEEE, 2011.

[23] L. Deng. Three classes of deep learning architectures and their applications: a tutorial survey. *APSIPA transactions on signal and information processing*, 2012.

[24] E. Dogantekin, M. Yilmaz, A. Dogantekin, E. Avci, and A. Sengur. A robust technique based on invariant moments - anfis for recognition of human parasite eggs in microscopic images. *Expert Systems with Applications*, 35(3):728–738, 2008.

[25] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013.

[26] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley & Sons, 2012.

[27] A. X. Falcão, J. Stolfi, and R. de Alencar Lotufo. The image foresting transform: Theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):19–29, 2004.

[28] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, and E. Culurciello. Hardware accelerated convolutional neural networks for synthetic vision systems. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 257–260, May 2010.

[29] R. Flores-Quispe, R. E. P. Escarcina, Y. Velazco-Paredes, and C. A. B. Castanon. Classification of human parasite eggs based on enhanced multitexton histogram. In *IEEE Colombian Conference on Communications and Computing (COLCOM)*, pages 1–6, 2014.

[30] P. Földiák. Sparse coding in the primate cortex. In *The Handbook of Brain Theory and Neural Networks*, pages 1064–1068. MIT Press, second edition, 2002. ISBN 0-262-01197-2.

[31] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015. URL http://arxiv.org/abs/1508.06576.

[32] B. Graham. Fractional max-pooling. *arXiv preprint arXiv:1412.6071*, 2014.

[33] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[34] R. H. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405(6789):947–951, 2000.

[35] Z. S. Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

[36] S. O. Haykin. *Neural Networks and Learning Machines*. Pearson Prentice Hall, 1999.

[37] W. Henneman, J. Sluimer, and e. a. Barnes, J. Hippocampal atrophy rates in alzheimer disease: Added value over whole brain volume measures. *Neurology*, 72(11):999–1007, 2009.

[38] P. Hensman and D. Masko. The impact of imbalanced training data for convolutional neural networks. Technical report, Royal Institute of Technology (KTH), Computer Science School, 2015. Degree project in Computer Science, First Level.

[39] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[40] G. E. Hinton and T. J. Sejnowski. Learning and releaming in boltzmann machines. *Parallel Distrilmted Processing*, 1, 1986.

[41] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[42] F. J. Huang and Y. LeCun. Large-scale learning with svm and convolutional for generic object categorization. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 284–291. IEEE, 2006.

[43] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. URL http://caffe.berkeleyvision.org/.

[44] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[45] A. Krizhevsky and G. Hinton. *Learning multiple layers of features from tiny images*. PhD thesis, University of Toronto, 2009.

[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1106–1114, 2012.

[47] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th International Conference on Computer Vision*, pages 365–372. IEEE, 2009.

[48] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face recognition: A convolutional neural-network approach. *Neural Networks, IEEE Transactions on*, 8(1): 98–113, 1997.

[49] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178. IEEE, 2006.

[50] Y. Lecun and C. Cortes. The MNIST database of handwritten digits, 2010. URL http://yann.lecun.com/exdb/mnist/.

[51] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[52] C.-Y. Lee, P. W. Gallagher, and Z. Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In *International Conference on Artificial Intelligence and Statistics*, 2016.

[53] Y. Leng, X. Xu, and G. Qi. Combining active learning and semi-supervised learning to construct svm classifier. *Knowledge-Based Systems*, 44:121–131, 2013.

[54] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[55] D. G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[56] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[57] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. R. Bach. Supervised dictionary learning. In *Advances in neural information processing systems*, pages 1033–1040, 2009.

[58] I. Masi, A. T. Tran, J. T. Leksut, T. Hassner, and G. Medioni. Do we really need to collect millions of faces for effective face recognition? *arXiv preprint arXiv:1603.07057*, 2016.

[59] D. Menotti, G. Chiachia, A. Pinto, W. Schwartz, H. Pedrini, A. X. Falcão, and A. Rocha. Deep representations for iris, face, and fingerprint spoofing detection. *IEEE Transactions on Information Forensics and Security (TIFs)*, PP(99):1–1, 2015.

[60] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[61] S. Moon and H.-L. Choi. Super resolution based on deep learning technique for constructing digital elevation model. In *AIAA SPACE 2016*, page 5608, 2016.

[62] A. Ng. Sparse autoencoder. *CS294A Lecture notes*, 72:1–19, 2011.

[63] S. O'Hara and B. A. Draper. Introduction to the bag of features paradigm for image classification and retrieval. *arXiv preprint arXiv:1101.3354*, 2011.

[64] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1717–1724, 2014.

[65] J. Papa. Classificação supervisionada de padrões utilizando floresta de caminhos otimos. *Biblioteca Digital da UNICAMP*, 2008.

[66] A. Peixinho, S. Martins, J. Vargas, A. Falcão, J. Gomes, and C. Suzuki. Diagnosis of human intestinal parasites by deep learning. In *Computational Vision and Medical Image Processing V: Proceedings of the 5th Eccomas Thematic Conference on Computational Vision and Medical Image Processing (VipIMAGE 2015, Tenerife, Spain, October 19-21, 2015)*, page 107. CRC Press, 2015.

[67] S. Peixoto, G. Cámara-Chávez, D. Menotti, G. Gonçalves, and W. Schwartz. Brazilian license plate character recognition using deep learning. In *Proc. of XI Workshop de Visão Computacional*, 2015.

[68] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1913–1921, 2015.

[69] N. Pinto, D. Doukhan, J. J. DiCarlo, and D. D. Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Comput Biol*, 5(11):e1000579, 2009.

[70] M. W. Popejoy. Working to overcome the global impact of neglected tropical diseases. *Perspectives in Public Health*, 132(4):192, 2012.

[71] G. Ramkumar, A. N. Swami, and H. America. Clustering data without distance functions. *IEEE Data Eng. Bull.*, 21(1):9–14, 1998.

[72] P. Rauber. *Visual Analytics Applied to Image Analysis: From Segmentation to Classification*. PhD thesis, University of Groningen and University of Campinas (co-title), 2017.

[73] P. Rauber, R. da Silva, S. Feringa, M. E. Celebi, A. Falcão, and A. Telea. Interactive image feature selection aided by dimensionality reduction. In *EuroVis Workshop on Visual Analytics (EuroVA)*, pages 19–23, Cagliari, Sardinia, May 2015. doi: 10.2312/eurova.20151098.

[74] P. Rauber, S. Fadel, A. Falcão, and A. C. Telea. Visualizing the hidden activity of artificial neural networks. *IEEE Transactions on Visualization and Computer Graphics (Proc. of the Visual Analytics Science and Technology 2016)*, 23(1):101–110, 2017.

[75] S. J. Raudys and A. K. Jain. Small sample size effects in statistical pattern recognition: recommendations for practitioners. *IEEE Transactions on pattern analysis and machine intelligence*, 13(3):252–264, 1991.

[76] R. Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[77] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[78] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[79] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115 (3):211–252, 2015.

[80] P. T. Saito, P. J. de Rezende, A. X. Falcão, C. T. Suzuki, and J. F. Gomes. An active learning paradigm based on a priori data reduction and organization. *Expert Systems with Applications*, 41(14):6086 – 6097, 2014. doi: http://dx.doi.org/10.1016/j.eswa.2014.04.007.

[81] P. T. Saito, C. T. Suzuki, J. F. Gomes, P. J. de Rezende, and A. X. Falcão. Robust active learning for the diagnosis of parasites. *Pattern Recognition*, 48(11):3572–3583, 2015.

[82] R. Salakhutdinov and G. E. Hinton. Deep boltzmann machines. In *International conference on artificial intelligence and statistics*, pages 448–455, 2009.

[83] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.

[84] J. Shi, S. Zhou, X. Liu, Q. Zhang, M. Lu, and T. Wang. Stacked deep polynomial network based representation learning for tumor classification with small ultrasound image dataset. *Neurocomputing*, 194:87–94, 2016.

[85] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE, 2003.

[86] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, DTIC Document, 1986.

[87] N. Sunderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford. On the performance of convnet features for place recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 4297–4304, Sept 2015.

[88] C. T. Suzuki, J. F. Gomes, A. X. Falcao, J. P. Papa, and S. Hoshino-Shimizu. Automatic segmentation and classification of human intestinal parasites from microscopy images. *Biomedical Engineering, IEEE Transactions ON*, 60(3):803–812, 2013.

[89] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5):1299–1312, May 2016. ISSN 0278-0062.

[90] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016. URL http://deeplearning.net/software/theano/.

[91] L. Torrey and J. Shavlik. Transfer learning. *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, 1:242, 2009.

[92] C.-F. Tsai. Bag-of-words representation in image annotation: A review. *ISRN Artificial Intelligence*, 2012:19 pages, 2012.

[93] J. Vargas, P. Saito, A. Falcão, P. de Rezende, and J. dos Santos. Superpixel-based interactive classification of very high resolution images. In *Proceedings of the 2014 27th SIBGRAPI Conference on Graphics, Patterns and Images*, pages 173–179, 2014.

[94] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

[95] Y. Xia. Fine-tuning for image style recognition. Technical report, Stanford University, 2014.

[96] C. Zhang, P. Zhou, C. Li, and L. Liu. A convolutional neural network for leaves recognition using data augmentation. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, pages 2143–2150. IEEE, 2015.

[97] S. Zhou, Q. Chen, and X. Wang. Active deep networks for semi-supervised sentiment classification. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1515–1523. Association for Computational Linguistics, 2010.

[98] S. Zhou, Q. Chen, and X. Wang. Active semi-supervised learning method with hybrid deep belief networks. *PloS one*, 9(9):e107122, 2014.

Appendix A

Active Learning Plots

A more detailed version of Active Learning plots, previously presented in Figure 4.5.

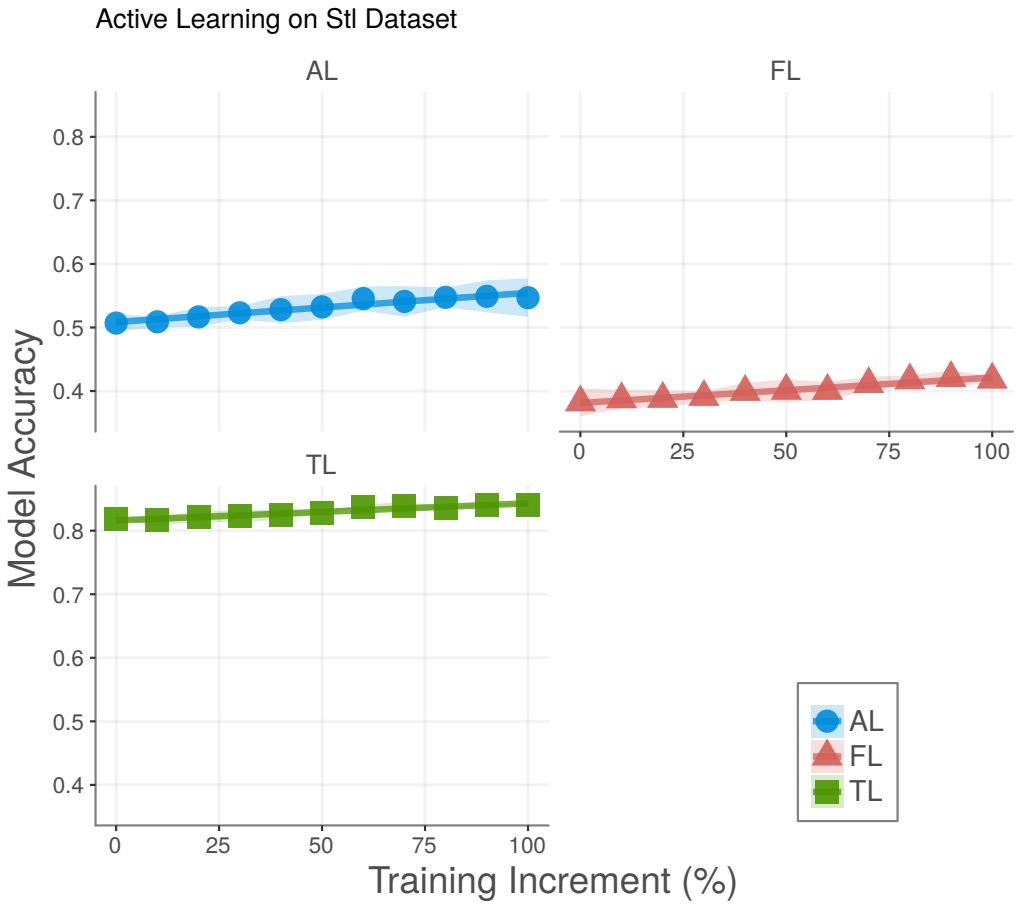


Figure A.1: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Stl10 data set.

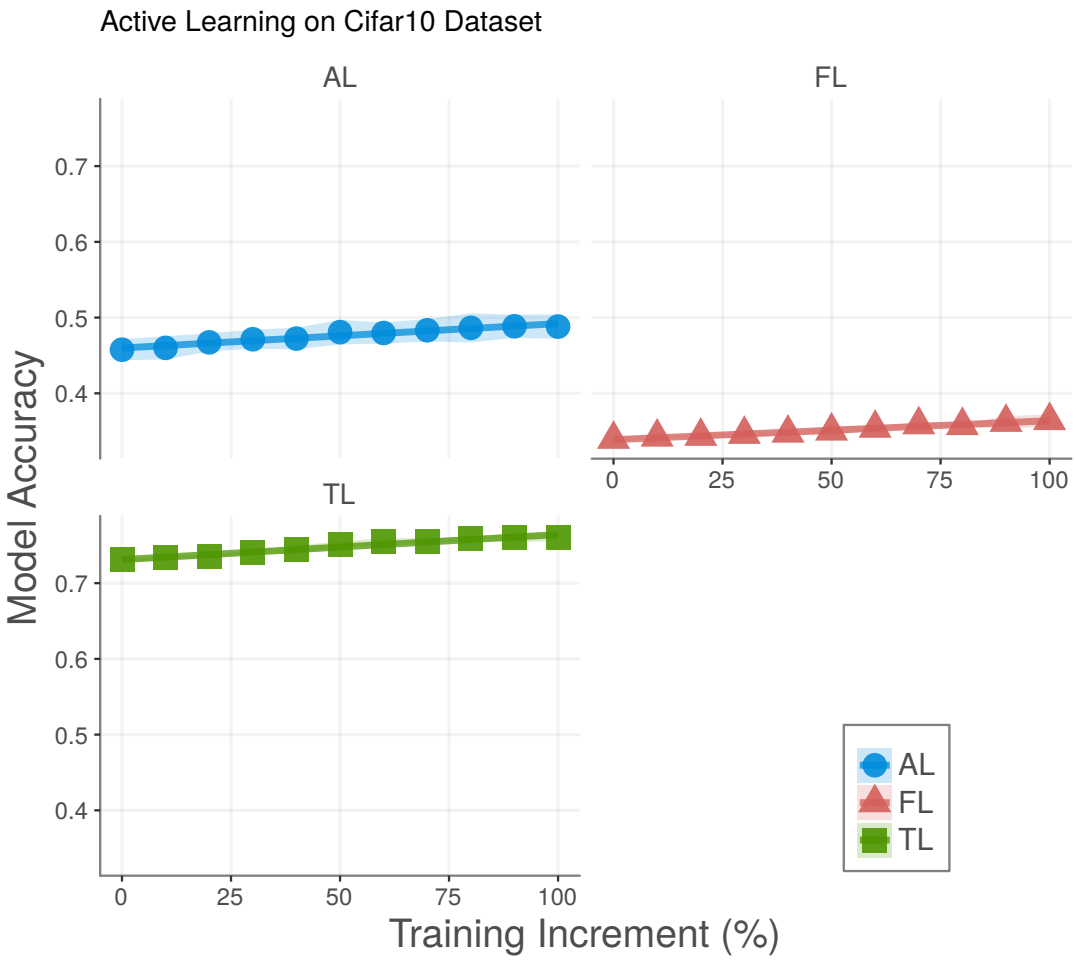


Figure A.2: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Cifar10 data set.

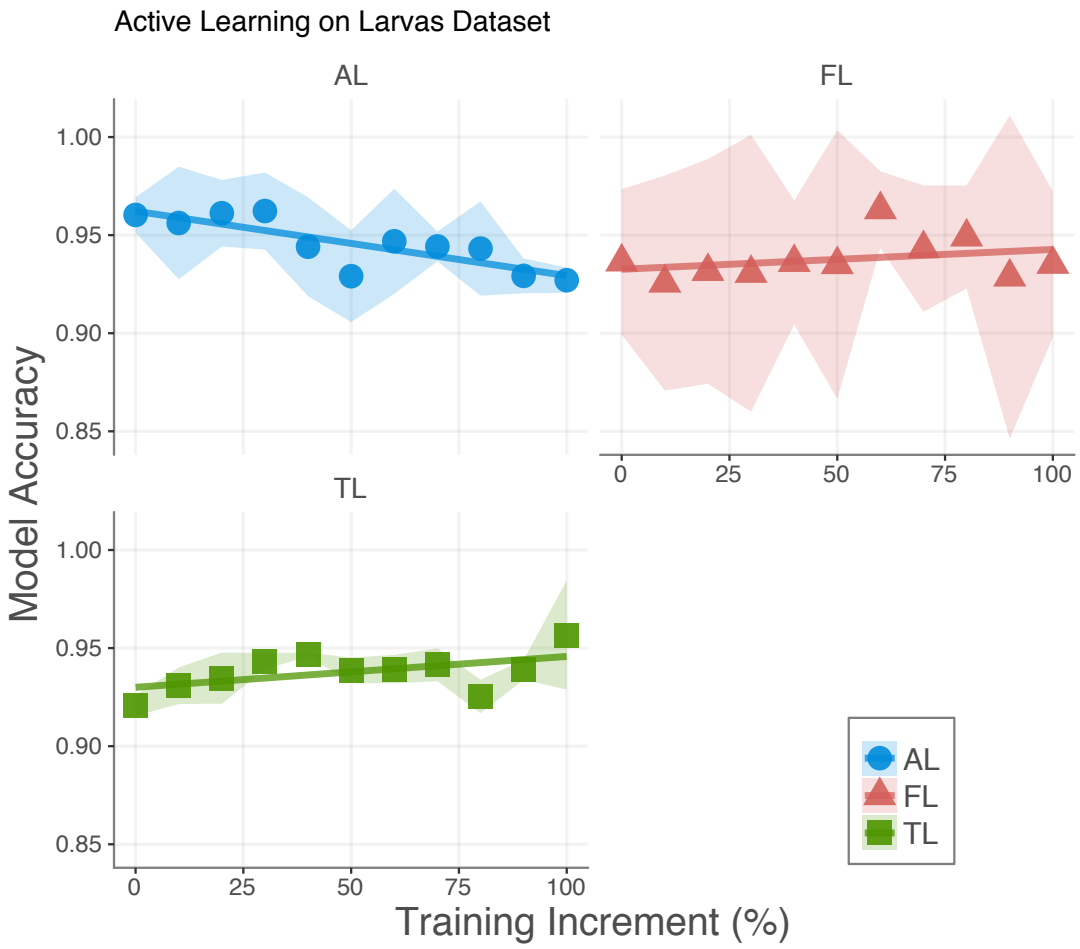


Figure A.3: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Larvae data set.

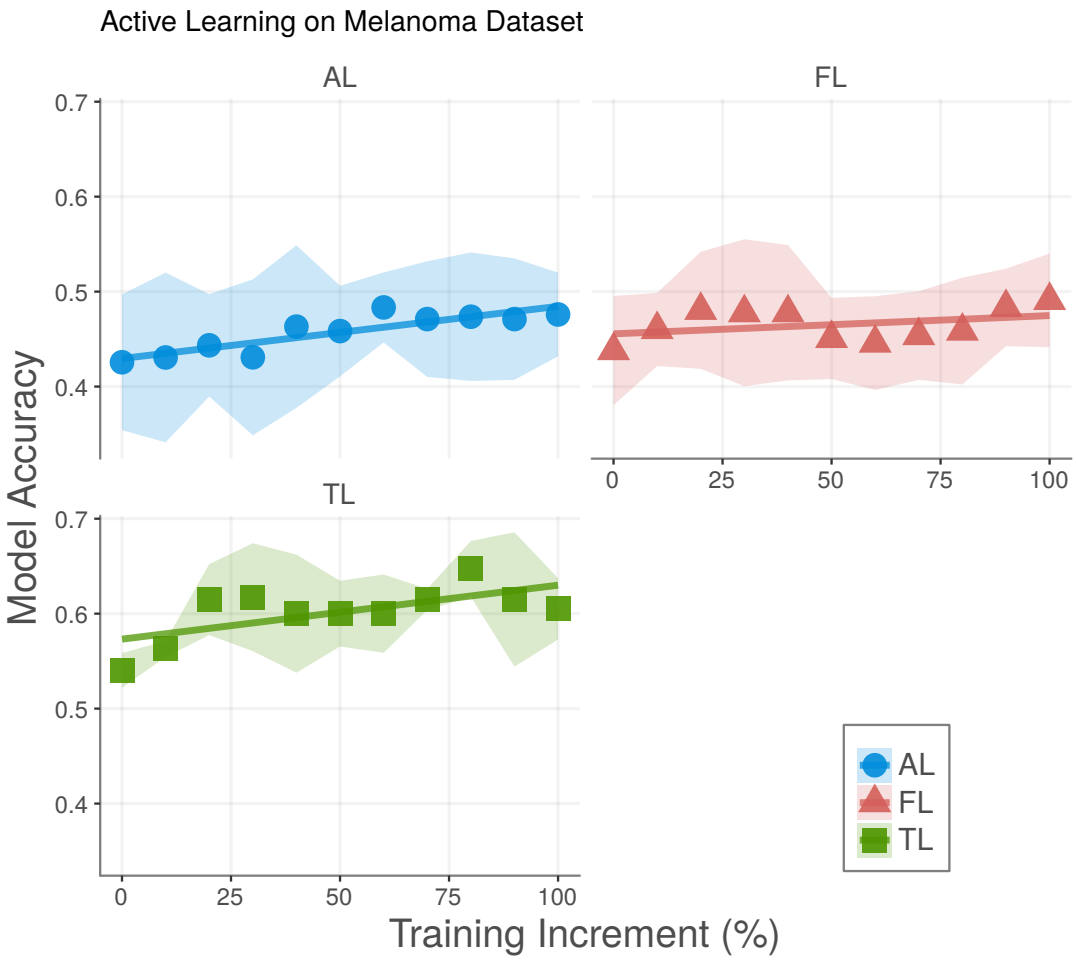


Figure A.4: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Melanoma data set.

Figure A.5: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Mnist data set.

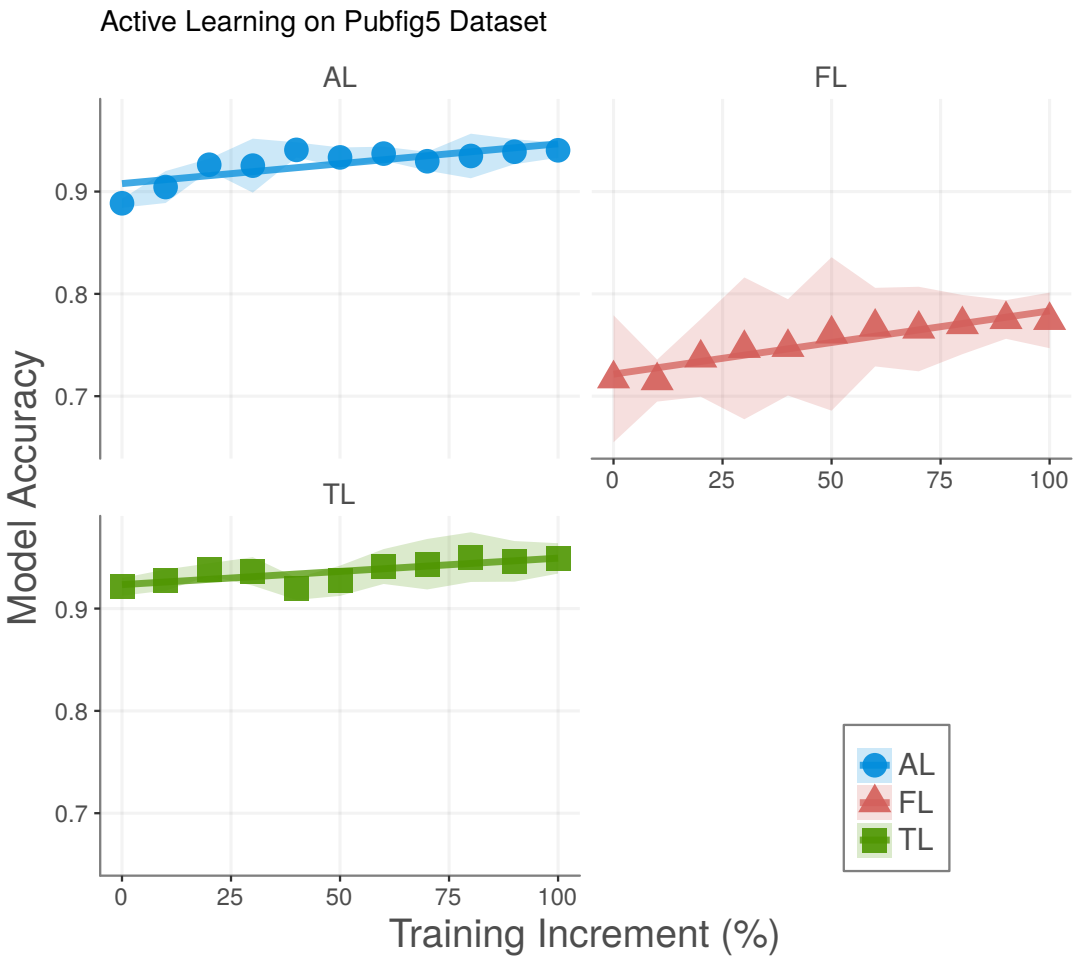


Figure A.6: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Pubfig5 data set.

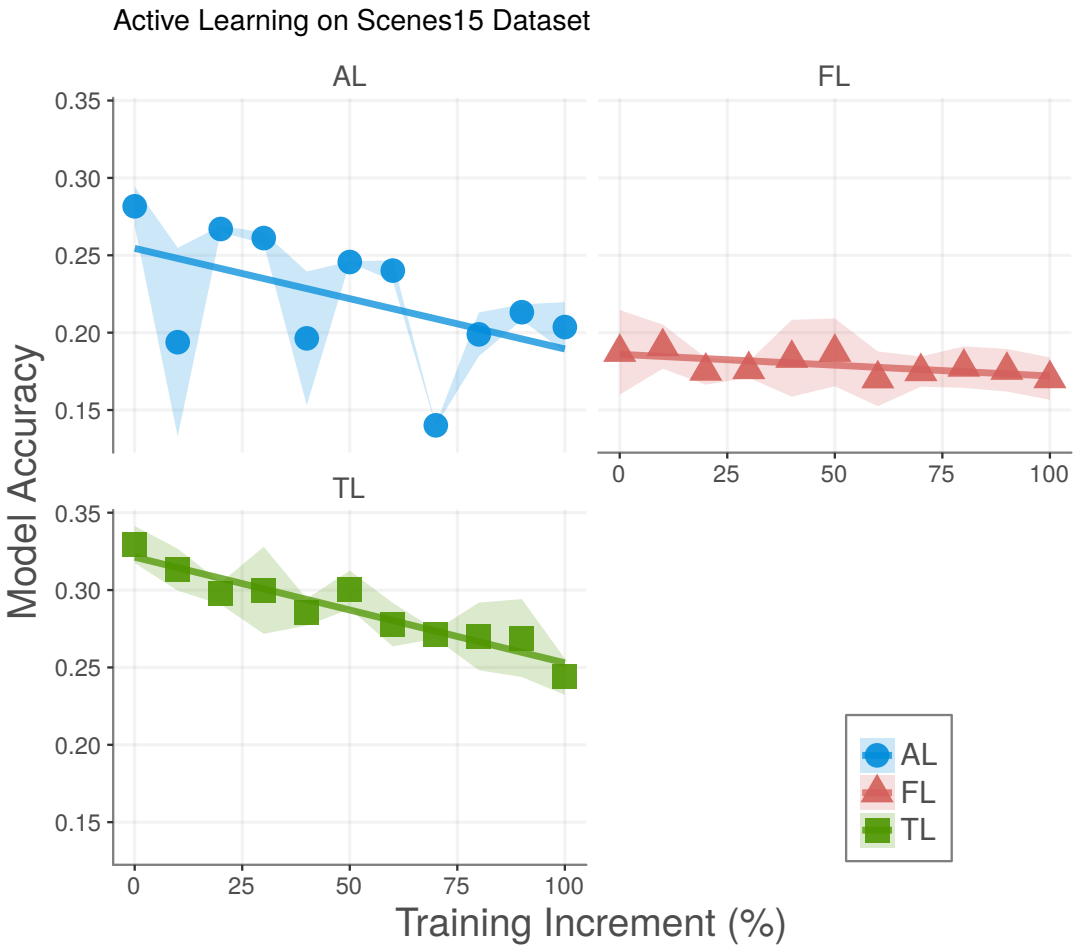


Figure A.7: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Scenes15 data set.

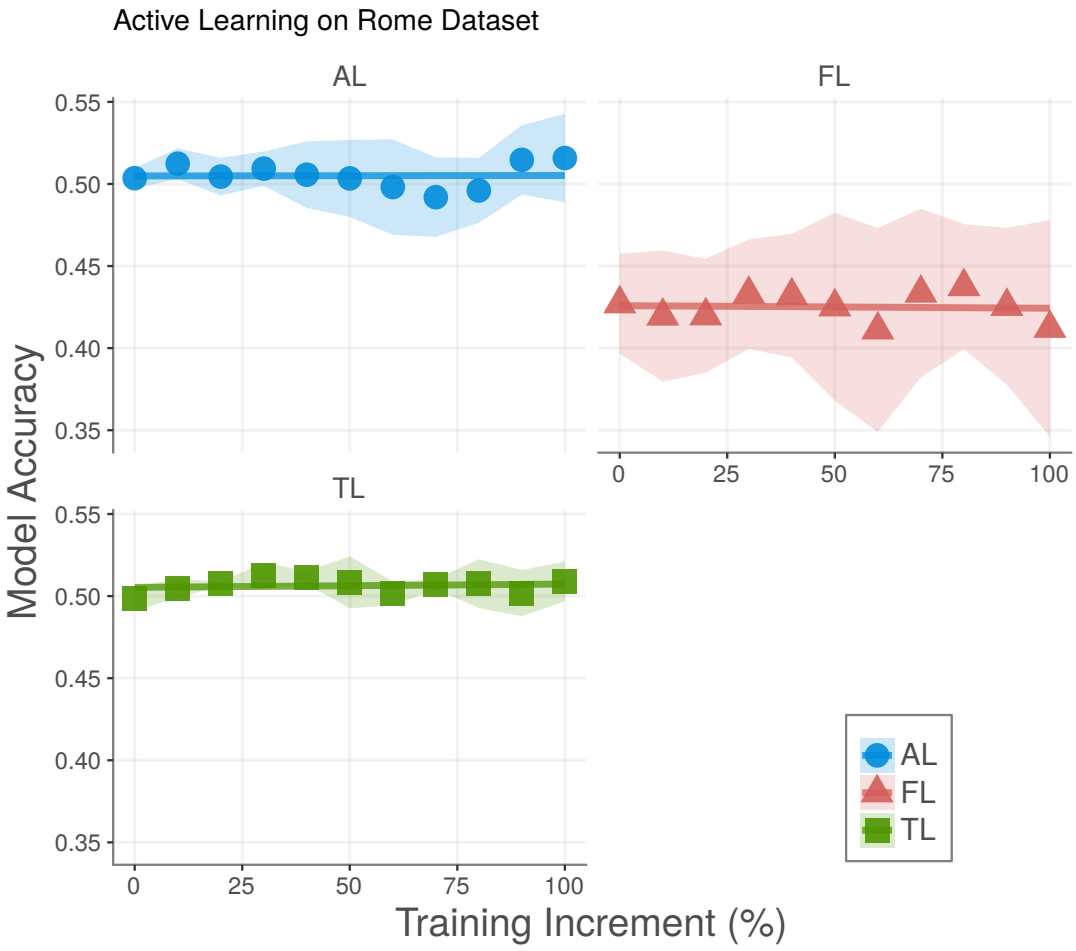


Figure A.8: Accuracy curves during active learning for image descriptors created by AL, FL, and TL on Rome data set.

Appendix B

Data Visualization Plots

A better version of the t-SNE 2D plot of data sets.

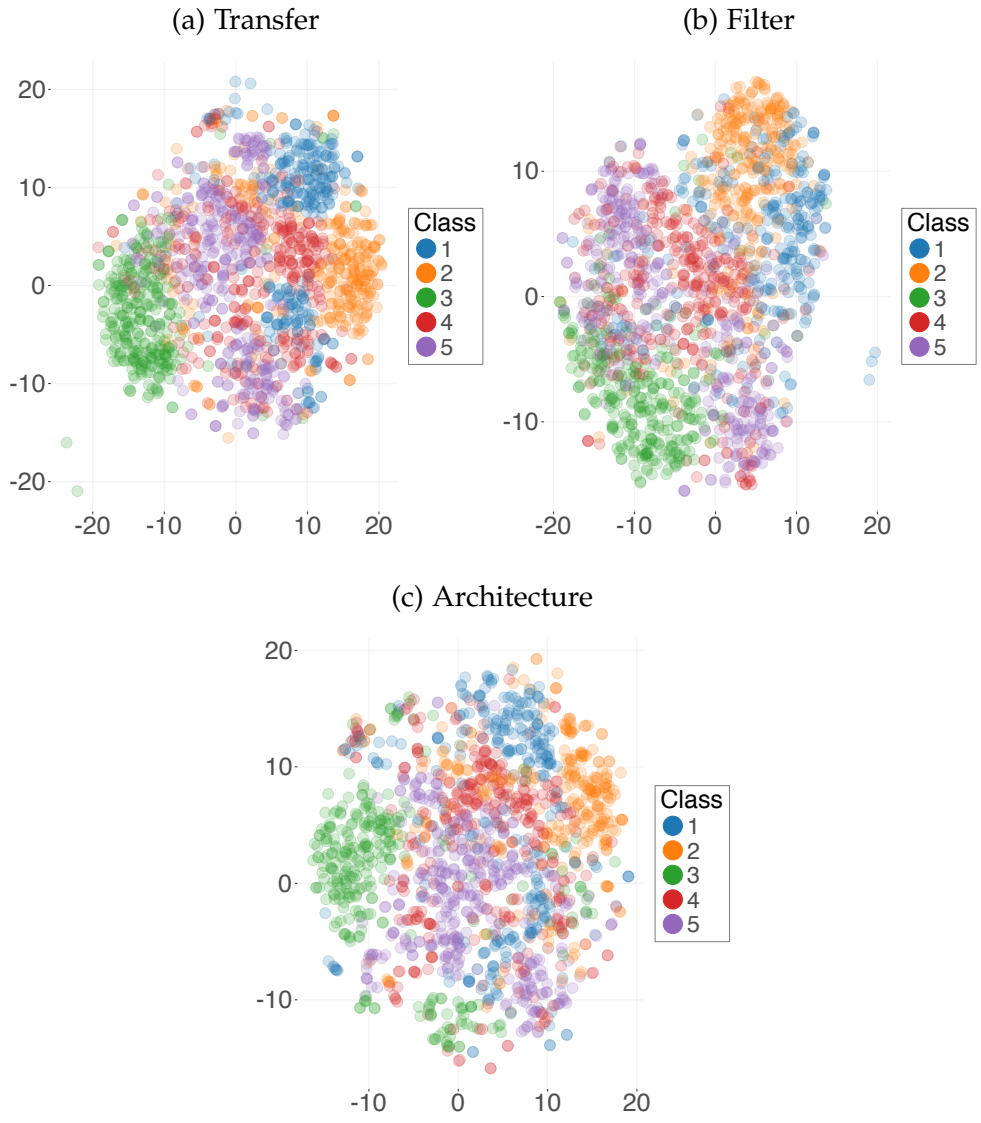


Figure B.1: t-SNE 2D visualization of all ConvNet descriptors on Pubfig5 dataset.

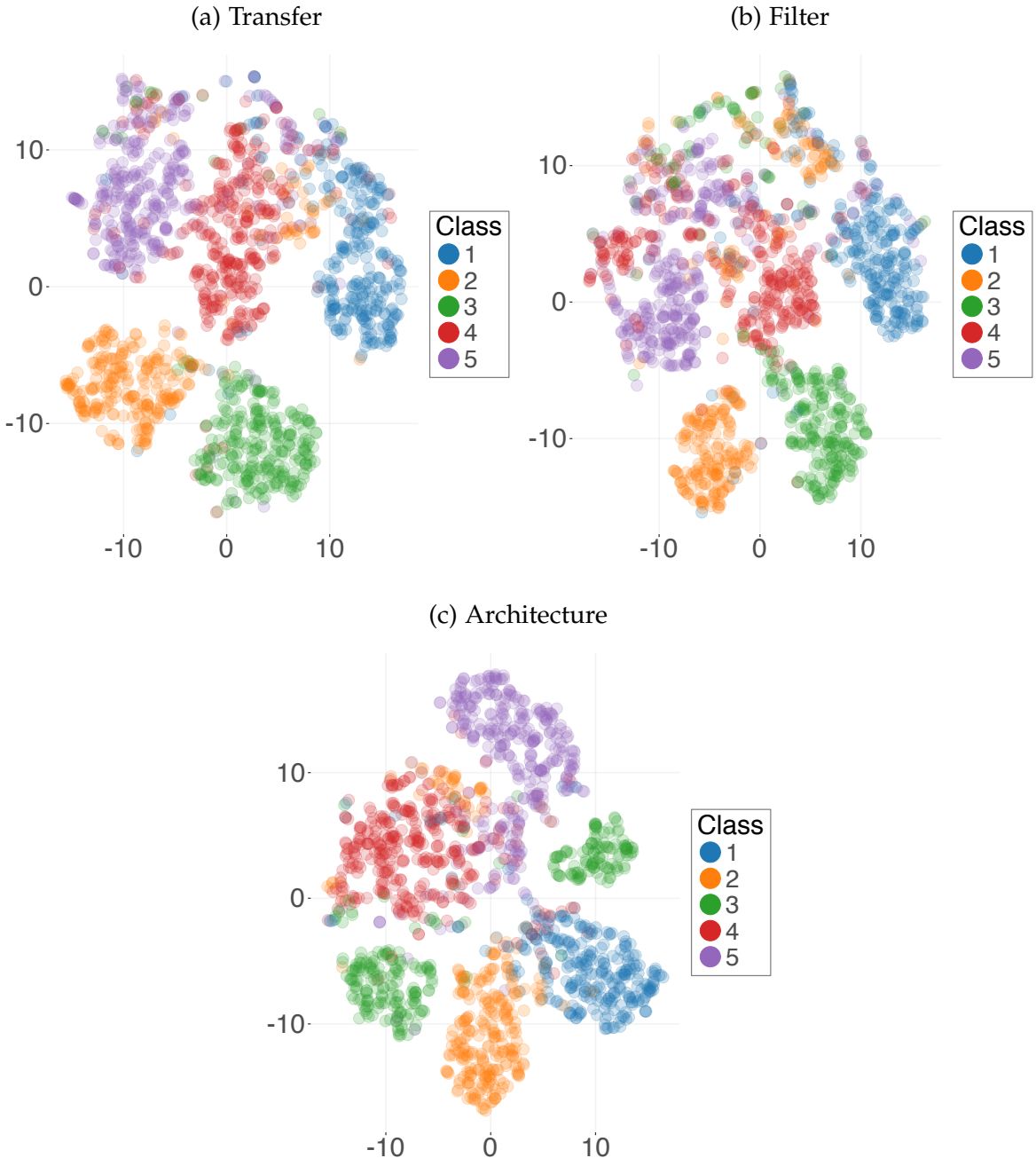


Figure B.2: t-SNE 2D visualization of the LDA dimensionality reduction of ConvNet descriptors on Pubfig5 dataset.

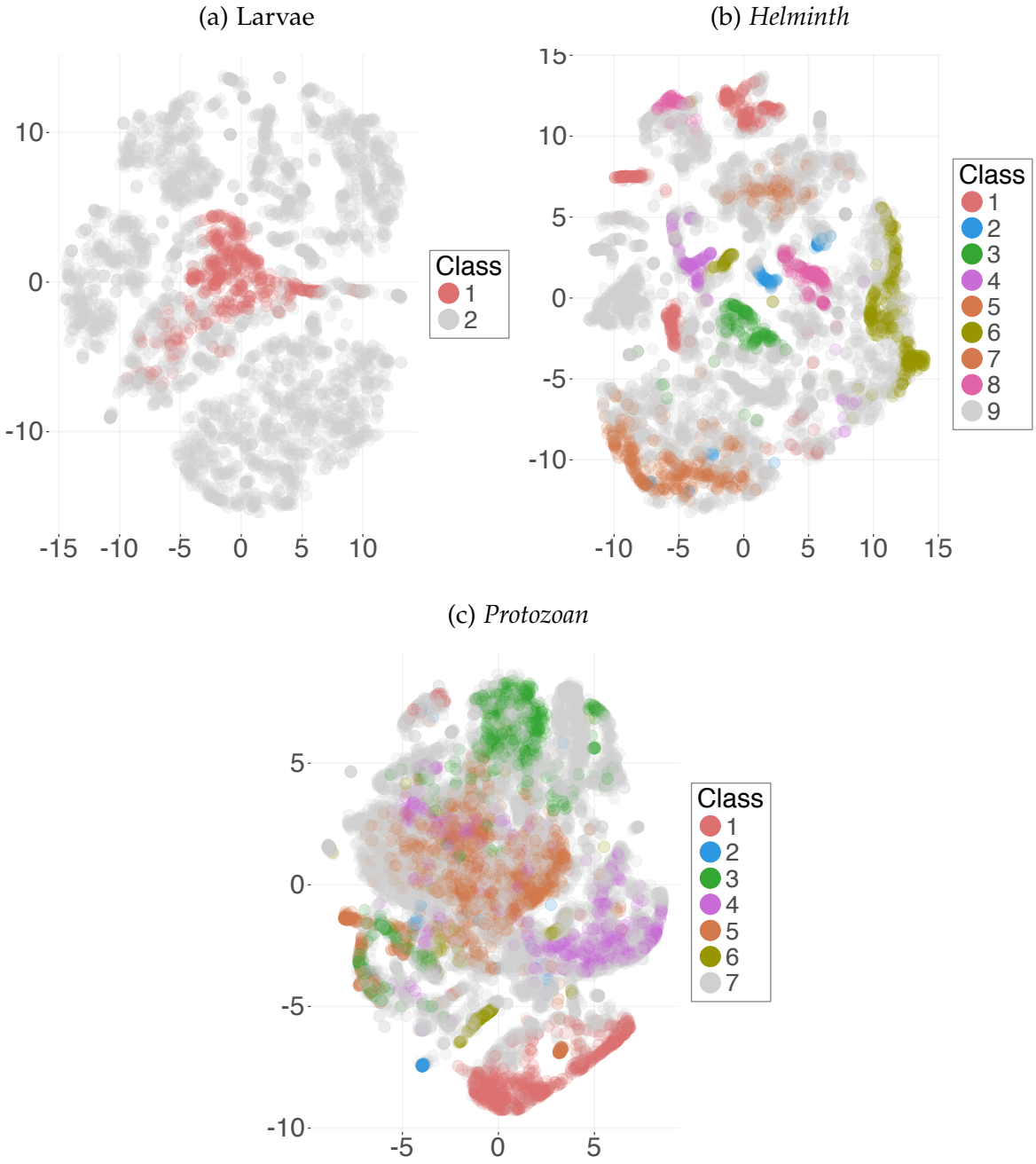


Figure B.3: The 2D projection of the samples in the feature space of the knowledge-based image descriptor [88], for each group of parasites.

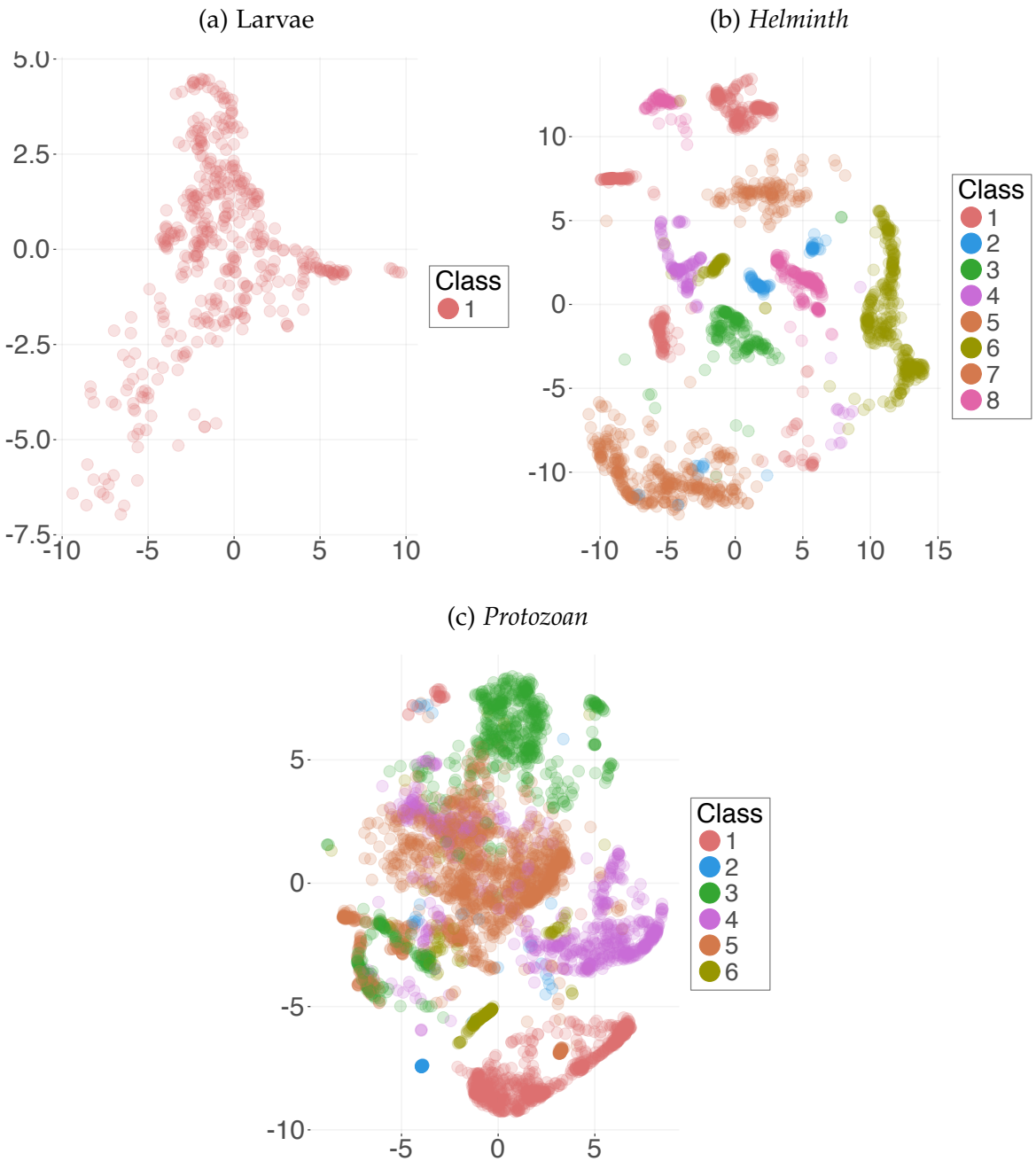


Figure B.4: t-SNE 2D visualization of the Suzuki et al. [88] descriptor, for all parasite groups, overlooking impurities, but **overlooking the fecal impurities**

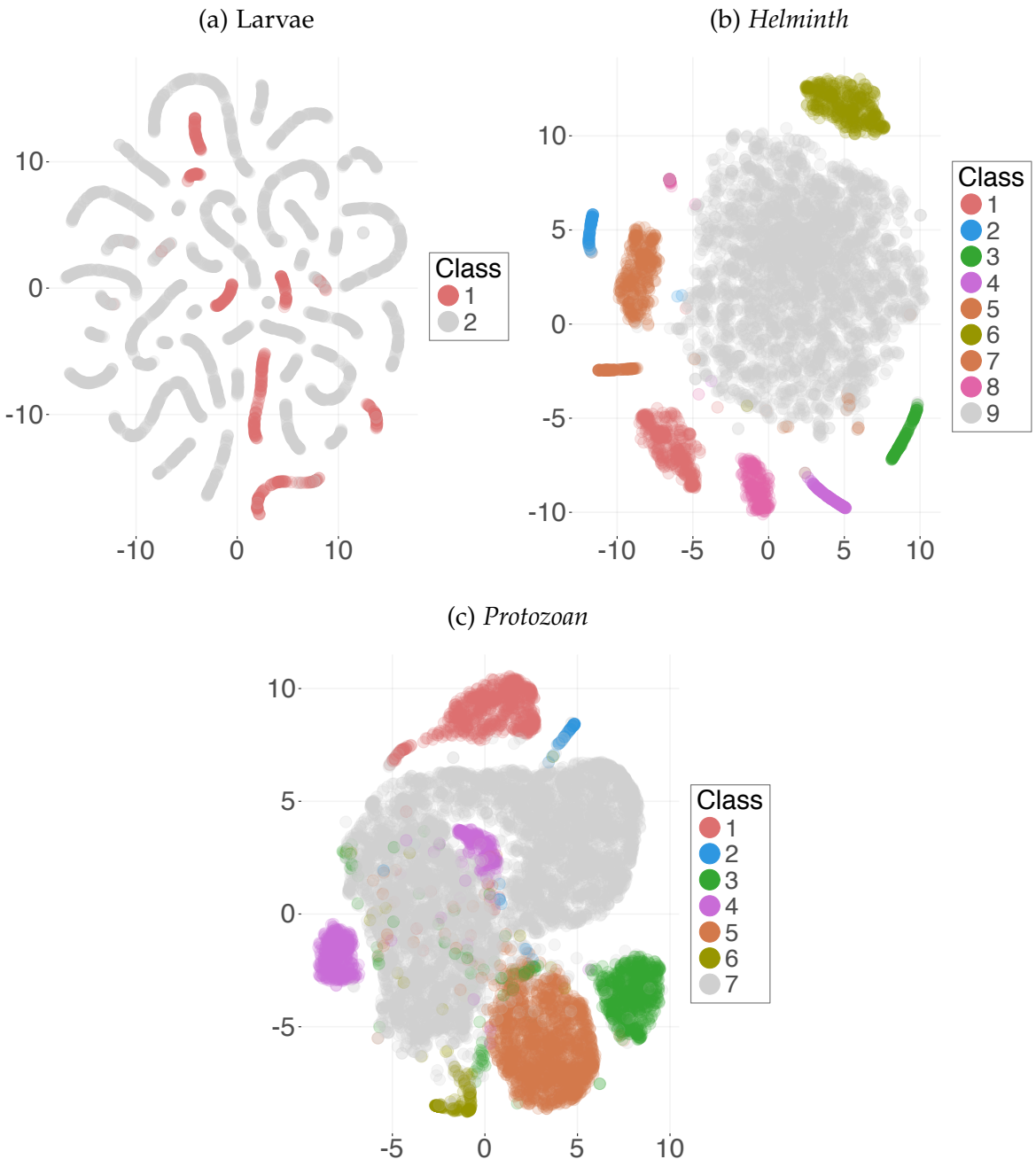


Figure B.5: The LDA followed by the 2D t-SNE projection of the samples in the feature space of the TL image descriptor, for each group of parasites.

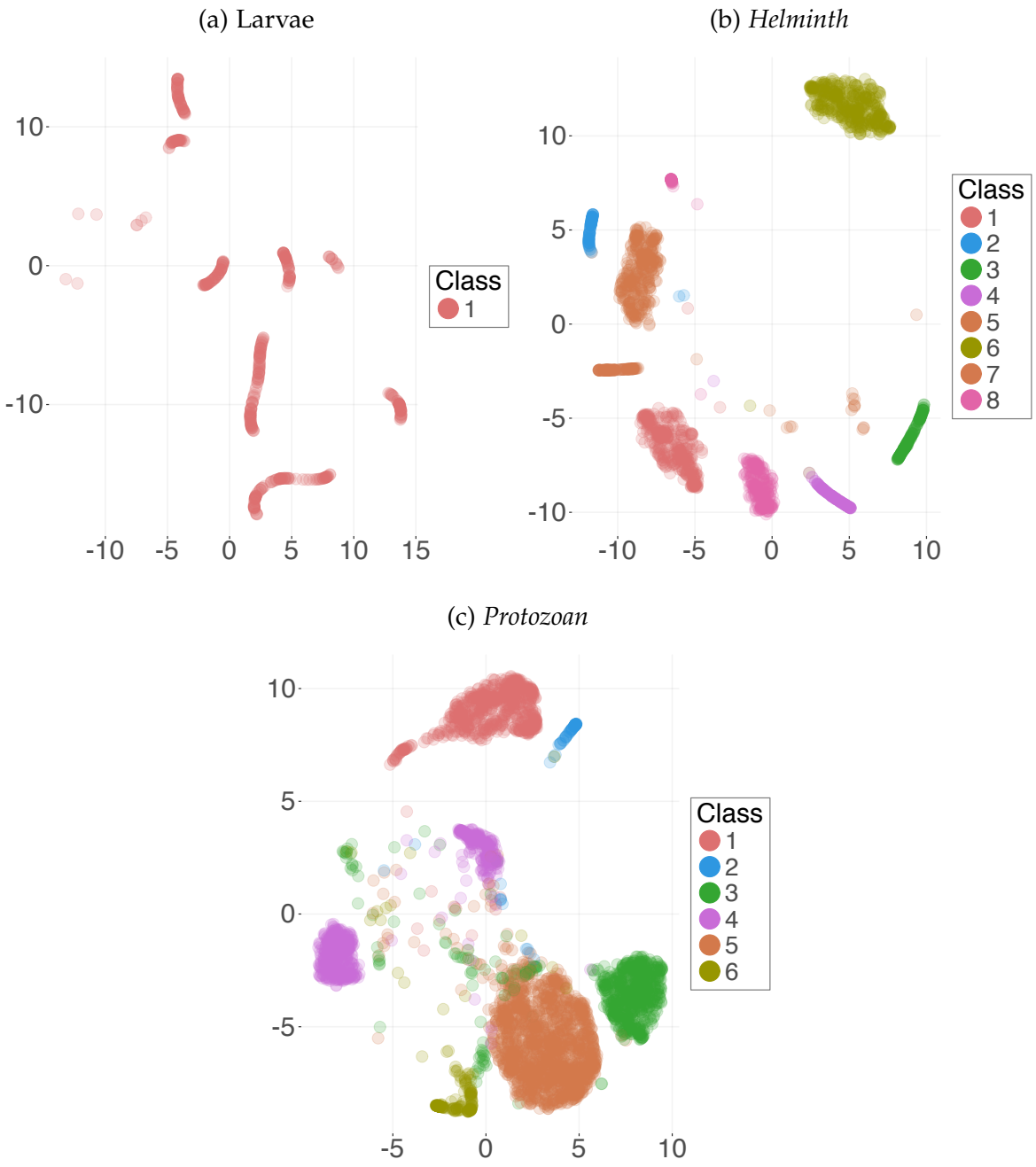


Figure B.6: The LDA followed by the 2D t-SNE projection of the samples in the feature space of the TL image descriptor, for each group of parasites, but **overlooking the fecal impurities**.