Deep learning for mining information from figures in biomedical literature

Almakky, Ibrahim

Award date:
2020

Awarding institution:
Coventry University

Deep Learning for Mining Information from Figures in Biomedical Literature

by

Ibrahim Almakky

September 2019

A thesis submitted in partial fulfilment of the University's requirements for the Degree of Doctor of Philosophy

Content removed on data protection grounds

Certificate of Ethical Approval

Applicant:

Ibrahim Almakky

Project Title:

Deep Learning for Mining Information from Figures in Biomedical Literature

This is to certify that the above named applicant has completed the Coventry University Ethical Approval process and their project has been confirmed and approved as Low Risk

Date of approval:

05 September 2019

Project Reference Number:

P93666


وقل رب زدني علما

Acknowledgements

First and foremost, my utmost gratitude goes to the Almighty, who has given me the strength to embark upon this journey. Followed by both my parents, Dr. Hala Dalbani and Dr. Muhammad Mouaffak Almakky, without whose support and encouragement I would not have been able to reach this stage. I am also deeply grateful to my brother Dr. Ahmed Al Makky, who has always set a great example for me to follow.

I would like to extend my sincere thanks to my director of studies, Dr. Vasile Palade, for his invaluable inputs and support throughout this journey. I would also like to thank Dr. Jianhua Yang, who was the one to help and encourage me to start this journey in the first place. Last but not least, Dr. Yih-Ling Hedley who has always been there to provide continuous support.

Thanks to Dr. Kamal Bentahar for his generous support and advice. This thanks also extends to the School of Computing, Electronics and Mathematics for providing the funding that made this research possible.

My special thanks goes to Dr. Romali Biswal, who has always encouraged me to persevere and push forward. Also, special thanks go to Dr. Ariel Ruiz-Garcia for the comprehensive discussions that often led to fruitful outcomes (fake news?). Many thanks to Luke Hicks, and hopefully the chicken-shish story will continue. And of course, thanks to Dr. Erik Barrow.

I would also like to thank all those who have been there with me on this journey, encouraging me, and providing me with the strength to complete this work. This includes the tea-time group: Muhammad Shamir, Anees Abu Monshar, Shirine El Zaatari, Mohamed Marei and Rohit Kshirsagar. This list also includes my cousins Omar, Hazar, Abed and Tarek.

Abstract

The biomedical field has witnessed an exponential growth over the past two decades due to technological leaps, such as next-generation sequencing, and rising concerns over healthcare and food security. Findings in biomedical research are mostly published in research articles, which are stored online in open-access databases. In such articles, researchers tend to use figures to illustrate and summarise some of the most important information concerning experimental settings and results. This information is often not retrievable from the articles' body of text, and therefore, methods have to be put in place to extract information from those figures. Such information can be later used towards the retrieval of the figures themselves or the articles that contain them.

This thesis explores the development of deep learning algorithms to facilitate the task of information extraction from biomedical figures. More specifically, the thesis focuses on the visual aspects of biomedical figures and what information can be extracted from the figure-image. With this goal in mind, this research investigates different aspects of representation learning and deep neural networks.

The thesis presents novel contributions, starting with a supervised deep representation learning method for classification. The development of this method is aimed at automatically extracting features that can enhance the classification performance of deep neural networks in general, and on biomedical figures in particular. Following that, a variety of deep learning approaches for the automatic extraction of visual features from biomedical figures were developed and put forward towards classification. Finally, a novel deep convolutional neural network was proposed to simplify the text localisation problem into a reconstruction one. With promising results for text localisation, text within biomedical figures can be extracted from the detected text regions and employed for figure indexing.

Table of contents

List of figures

List of tables

Nomenclature

Acronyms / Abbreviations

AE Autoencoder

DCNN Deep Convolutional Neural Network

LDA Linear Discriminant Analysis

MSE Mean Squared Error

OCR Optical Character Recognition

PDF Portable Document Format

ReLU Rectified Linear Unit

ResNet Residual Neural Network

SDAE Stacked Deep Autoencoder

SVM Support Vector Machine

t-SNE T-Stochastic Neighbour Embedding

Chapter 1

Introduction

1.1 Motivation

The corpus of scholarly articles has been exponentially growing and adding upon an already large database of published works. This growth has been especially apparent in the biomedical field, where there has also been an increasing push towards open-access online databases. This rapid growth has rendered researchers in the field unable to keep up with its development, making it imperative to put systems in place that would enable easier search of relevant articles. However, search engines that have been put in place to do this have focused on the articles' bodies of text while ignoring information stored in figures. Figures are important sources of information in all fields of research, however, within the biomedical field they tend to contain some of the most important information surrounding the experimental settings and results. Currently, figure indexing is constrained to the text contained within their captions, which often does not provide sufficient information. Therefore, it has become essential for retrieval systems to have the ability to function beyond figure captions, tapping into the visual and textual features included within the figure itself.

1.2 Figure Mining

The Oxford dictionary defines a figure as "A diagram or illustrative drawing, especially in a book or magazine." [1]. Figures are used extensively in scholarly articles to communicate large amounts of complex information that would be difficult to explain in plain text. This makes figures a unique type of images that contain information with varying complexities, where a figure can simply be an image of a thing or it can get more complex, such as data plots. Information within figures can be stored either in visual form (e.g. CT scan) or in textual form (e.g. gene sequence) or in a combination of textual and visual forms (e.g. pie chart). Figures in published works are accompanied with captions that try to provide a brief description of what the figure contains. Additionally, the caption includes the index of the figures, which is used to refer to a specific figure within the body of text. Furthermore, figures can contain multiple subfigures, each with its own sub-caption, those being referred to as compound figures.

Indexing scientific articles is based on keywords that exist within the articles' body of text. On the other hand, indexing figures has been limited to the caption texts rather the figure content, because of their complex format. The preferred publishing format for scientific articles is the portable document format (PDF) and figures are mostly included as images within those files. Setting aside the fact that the visual representations contained within the figure are not simply indexable, even the text contained within the figure is ignored because it is often just encoded within the image pixels and not as text within the PDF. As the information included within figures is often not included anywhere else in the published work, it becomes imperative to bring forth new methods to extract information from such figures that would provide more effective indexing.

The figure mining process starts with extracting the figure along with its caption from the containing document [2, 3]. This step could also involve the extraction of texts associated with the figure from the body of text, where the figure is mentioned [4]. The type of class is then determined through a classification process that can use visual or textual features or a combination of the two. Finally, optical character recognition (OCR) is used to extract the text contained within the figure. The mining process may also include a compound figure separation stage, if the extracted figure was a compound figure [5].

1.2.1 Figure Extraction

Figure extraction is the initial stage of figure mining, where the figure located within the containing document is extracted into a useful format. With the current dominance of the PDF format over the publication formats of scholarly articles, most of the focus has been on extracting figures from PDF files [2]. However, recently the rise of open-access online publications has significantly simplified the figure extraction task.

PDF files are encoded using a series of operators, with each operator responsible for drawing specific elements in the document. There are three main operators responsible for displaying the following objects at specific coordinates:

- Characters in specific styles and fonts;
- Vector graphics, such as lines and various geometries;
- Images embedded internally within the document.

In this manner, figures within a PDF document can be encoded as follows:

- Entirely embedded in the file as a single image;
- A collection of images arranged together;
- A collection of vector graphics and text;
- A combination of the three operators.

There exist many off-the-shelf tools, such as PDFBox [6], Poppler [7] and Xpdf [8], capable of extracting images from PDF files. However, unlike the figures composed of text and vector graphics, the contents of image encoded figures require further processing to extract the text contained within them. This thesis focuses on the first two types of figures, which are contained within the published work, either as a single image or as a collection of images.

These PDF manipulation tools have been used for different approaches to extract figures and their metadata. Xpdf was used by Lopez et al. [3] to develop a system to extract figures from biomedical literature. Using Xpdf to tap into the PDF specification, their aim was to extract subfigures and captions, while also identifying the figure that they belong to. They

also tried to filter out logo images, which are sometimes included within the PDF documents for scholarly articles. Similarly, Choudhury et al. [4] employed PDFBox to create a system designed to extract figure captions and associate them with their figures. Unlike Lopez et al. [3], Choudhury et al. [4] considered vector graphics figures and incorporated a machine learning aspect to their system. With a completely rule-based approach, Clark and Divvala [2] developed a similar system to that developed by Lopez et al. [3]. However, they used Poppler instead of Xpdf to extract the different elements from the PDF files.

This thesis focuses on the tasks following figure extraction, where the figure and its caption have been extracted from the containing document. This was motivated by the fact that more and more publishers are releasing a webpage format as well as a PDF document for the scholarly article.

1.2.2 Compound Figures

Compound figures are figures that contain two or more subfigures, either of the same type of different types. Figure 1.1 shows an example of a compound figure that contains subfigures of various types. Separating compound figures into their constituent subfigures is a crucial task. However, to achieve that, we first have to identify whether a figure is compound or not. This step was ignored by many researchers who have endeavoured the figure separation task [9]. Delving into this problem, Antani et al. [10] attempted both, detection and decomposition of compound figures. However, before starting any of the two tasks, they tried to estimate the number of panels within the figures by processing the figure captions using natural language processing (NLP) methods. This prediction is prone to errors considering that figures might not have captions associated with them or even be associated with the wrong caption. Binarizing the image was their next step, which they used to compute both horizontal and vertical profiles in search for evidence of white or black lines that are less than or equal to 5% of the total image width or height. Following a similar model, Lopez et al. [11] also tried to predict the number of panels in figures before segmenting them. Conceptually, this would be a good idea to validate the segmentation results, if the method yields accurate predictions. However, Lopez et al. did not solely depend on captions for this prediction but they also utilised the number of labels, connected components and sub-captions.

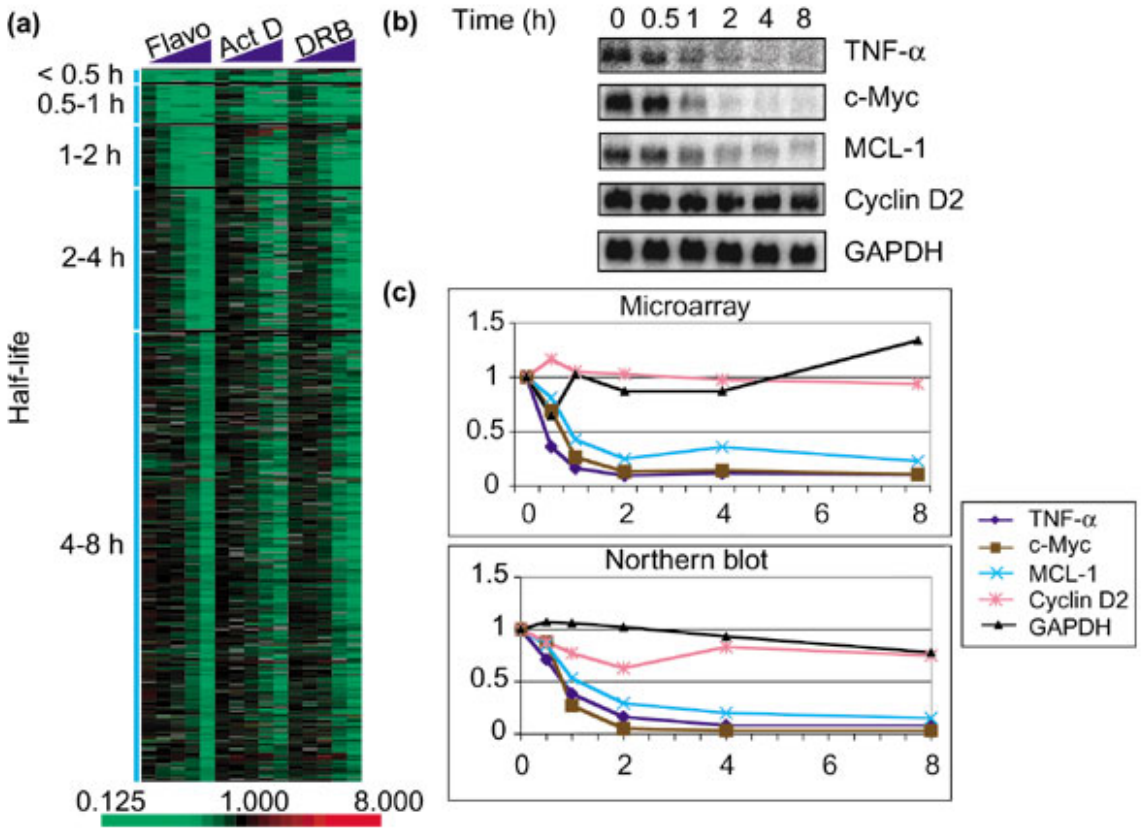


Fig. 1.1 An example of a compound biomedical figure [12].

1.3 Mining Information from Biomedical Literature

Biomedical literature has been rapidly growing with thousands of articles published every day, thus making it difficult for researchers to match its pace and keep up with its developments. In response to this, many systems have emerged to facilitate researchers' access to information at a matching pace, such as the Yale image finder [13]. However, such systems are blind to some of the most crucial information within biomedical articles surrounding experimental settings and results, which is contained within figures.

In an effort towards the extraction of information from biomedical figures that can be used for indexing, this thesis addresses the current challenges of biomedical figure mining. This is demonstrated in Figure 1.2, which shows the main stages involved in this research towards the mining of information from biomedical figures. Starting from figures extracted from biomedical research articles and ending with the figure index. This happens through

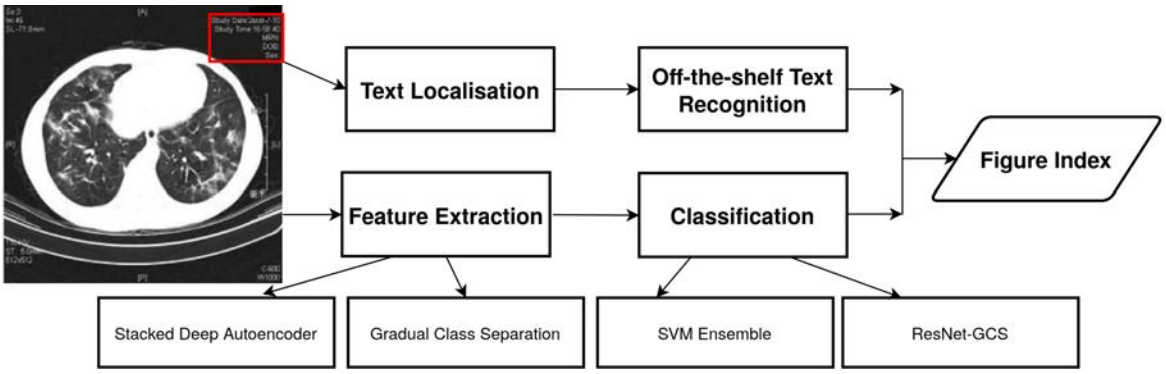


Fig. 1.2 Flowchart of the various methods developed in this thesis towards figure mining.

two different channels, one that addresses the textual features through text localisation and recognition and another that addresses visual features through automatic feature extraction and image classification techniques.

1.4 Research Aim and Objectives

With the exponential increase in computing power and GPU processing speeds, the deep learning field has developed quickly to even surpass human abilities on computer vision tasks [14] among others [15]. This makes it a logical choice to deal with mining information from biomedical figures, considering their complex formats. Therefore, this thesis aims to develop various deep learning approaches towards the extraction of information from biomedical figure images, while setting out to accomplish the following objectives:

- To develop deep learning approaches to automatically extract features from biomedical figures, which could be used towards classification or clustering of such figures (Chapter 3).
- To develop image classification methods that would enhance the classification of biomedical figures into categories. This is to help search engines with indexing figure contents, and therefore lead to the retrieval of the paper containing the figure or even the figure itself (Chapter 4).
- To localise the text within biomedical figures to facilitate the text extraction process (Chapter 5).

1.5 Thesis Contributions

The work presented in this thesis offers the following original contributions:

- A novel effective representation learning method is proposed (Chapter 3), which is used to enhance the classification of biomedical figures in Chapter 4.
- A new approach to biomedical figure classification is devised to overcome the specific challenges in biomedical figure classification (Chapter 4).
- A novel deep learning architecture is proposed for text localisation within images in general, and within biomedical figures in particular (Chapter 5).

1.6 Research Outputs

As a result of the research conducted for this PhD thesis, the following articles have been published:

- I. Almakky, V. Palade, Y. Hedley, and J. Yang, "A stacked deep autoencoder model for biomedical figure classification", in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 2018, pp. 1134–1138. Apr. 2018.
- I. Almakky, V. Palade, and A. Ruiz-Garcia, "Deep Convolutional Neural Networks for Text Localisation in Biomedical Literature Figures," in 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1–5, IEEE, July 2019.

1.7 Thesis Overview

Starting with the background and literature review in Chapter 2, this thesis first delves into the research that has been done in the relevant areas. Chapter 2 also compares and critically evaluates the previous approaches to retrieving information from figure images in the biomedical fields as well as other fields. Later in Chapter 3, a novel approach to learning data representations effectively is presented. Demonstrating the effectiveness of this method,

the chapter then presents experiments on different popular benchmark datasets tested using different deep architectures. This method then ties into the next two chapters, where it is used to extract meaningful features that are used for classification, and it is then compared to some other approaches. Chapter 4 then investigates the figure image classification to extract the first piece of information from the puzzle. It looks closely at the specific challenges with the classification of biomedical figure images, while offering different approaches to solving them. The chapter also presents and compares the results of the different approaches, and finally concludes with the strengths and weaknesses of each method. Chapter 5 follows with acquiring the next piece of the puzzle, the text contained within the figure images; starting with the description of the specific challenges to extracting text from biomedical figures when compared with documents and natural scene images. Following that, a novel method is put forward to localise the text within the figure image and experiments that demonstrate its performance on biomedical figures. The thesis finally concludes in Chapter 6, where the outcomes of the different chapters are brought together and the future challenges are presented.

Chapter 2

Background and Literature Review

2.1 Introduction

This chapter provides a theoretical background of the underlying research behind the different stages of mining information from figures in biomedical literature. As discussed earlier in Chapter 1 and demonstrated in Figure 1.2, mining information from biomedical figures is approached through both the visual and textual aspects of the figure. More specifically, image classification techniques can be used with the visual aspects and OCR techniques can be used with the textual ones.

In such a manner, the chapter first defines the figure classification task and discusses the traditional methods that have been used to classify biomedical figures. Following that, the chapter delves into more recent deep learning, automatic feature extraction and representation learning methods that could be used towards biomedical figure classification. The chapter then presents different classification methods that form the basis of the approaches developed in this thesis.

Aside from figure classification, the chapter analyses the different text extraction methods that have been employed in the past to extract text from biomedical figures. The chapter also discusses more recent deep learning efforts that have been effective in extracting text from natural scene images, which have the potential of being used towards extracting text from biomedical figures. Finally, the chapter concludes by identifying the current gaps in research that would enable better mining of information from biomedical figures.

2.2 Biomedical Figure Classification

Figure classification is the task of determining the category that a figure belongs to. This task is not exclusive to the biomedical field [16]; however, biomedical figures have gained extra attention due to their ubiquitous nature in biomedical literature, as well as the importance of the information contained within them. Figure classification in general is aimed at extracting an essential piece of information that can help with indexing the figure and with any further steps that need to be carried out to extract additional information. Research into figure classification has been focused on supervised learning with hand-crafted features, which can be either visual [17, 18], textual [19] or a combination of the two [20–24]. However, some more recent attempts have used deep learning approaches to classify biomedical figures [25, 26].

2.2.1 Taxonomy of Figures in Biomedical Literature

Before going into the feature engineering and classification methods that have been used to classify biomedical figures, it is important to discuss the taxonomy of figures in biomedical literature. The taxonomy of biomedical figures has been taking shape over the years and it has been getting more and more concise. In 2006, Rafkind et al. [20] first proposed a taxonomy of biomedical figures that divided biomedical literature figures into five classes: gel-image, graph, image-of-thing, mix and model. The small number of classes in this taxonomy and their broad nature makes them less effective to be used towards figure indexing.

This meant that for taxonomies to be more effective, classes had to be further divided into sub-classes. In such way Shatkay et al. [17] proposed the first hierarchical taxonomy that divided biomedical figures first into three main categories: graphical, experimental and other. The first two classes were then further divided into sub-classes, where the class of graphical figures was split into line charts, bar charts and other diagrams. As for the class of experimental figures, it was divided into gel electrophoresis, fluorescence microscopy and other types of microscopy. Later on, Lopez et al. [27] expanded on the hierarchical taxonomy suggested by Shatkay et al. [17] through adding another main class for text containing figures. The new text figures class was then subdivided into sequence and text figures. In addition to line and bar chart figure classes under the graph class, Lopez et al. also added 3D

models. They also substituted the gel electrophoresis with gel/blot/autoradiography, plate and microscopy classes. Muller et al. [28] proposed a more comprehensive taxonomy that included three levels of hierarchy. This taxonomy was adapted by the ImageCLEF medical challenge [29] for biomedical figure classification in 2013, which is described further in Chapter 4.

2.2.2 Differentiating Features

Prior the rise of deep learning, many efforts into the classification of biomedical figures were using hand-engineered features towards classification [17–24]. As detailed in this section, features were extracted either from the image of the figure (visual features) or from the text contained in the figure caption (textual features).

Visual Features

Due to the visual complexity of biomedical figures and the wide range of biomedical figure classes, it has been challenging to find a set of visual features that are capable of differentiating between the different classes. Following are some of the main visual features that were used as part of different classification approaches:

- **Intensity histograms.** Rafkind et al. [20] utilised a normalised 256-bin grey-scale intensity histogram to use the entropy, mean, 2nd and 3rd moments as features towards classification. Similarly, Shatkay et al. [17] used the mean, variance and skewness as well as features from grey-level pixel-value histograms. Later, Han and Chen [21] and Kim et al. [22] went a step further with adding colour intensity histograms in addition to the grey intensity histogram.
- **Edge direction histograms.** Rafkind et al. [20] hypothesized that edge direction histograms (EDH) can help differentiate figures that predominately contain straight lines, such as charts and graphs. EDH was calculated after convolving the grey-scaled image into 3×3 blocks using the Sobel edge operators. A similar approach using the Sobel operators was also carried out by Kim et al. [22]. Han and Chen [21] on the other hand, adopted a block-based edge histogram. Shatkay et al. [17] used Canny's

edge detector to detect edges from the figure images to then formulate a histogram from edges that share the same direction.

- **Edge-based axis features.** From the same motivation as above, Rafkind et al. [20] also explored edge-based axis features by first extracting Sobel edges from the grey-scaled figures. The vertical and horizontal sums were then captured from the resulting binary image resulting into two vectors. The vectors were then normalised and the entropy, mean, 2nd and 3rd moments were used as features from each axis.

- **Variance histograms.** Han and Chen [21] formed block-based variance histograms to capture the local variation of the pixel values in the image.

- **Bag of visual words.** Han and Chen [21] also used grid-sampling patched and then calculated appearance-based descriptors for the different patches. Using a modified Scale Invariant Feature Transform (SIFT) method [30], they represented every patch of the image using a set of features.

- **Bag of colours.** Based on the bag of visual words representation of images, de Herrera et al. [18] proposed a method to extract colour signatures of biomedical figures. More specifically, each image was visually summarised using the bag of colours from a pre-set vocabulary of colours formulated from a subset of the dataset.

- **Skew difference.** The skew difference was used by Kim et al. [22] to identify the "Model" figures, which they had defined for their taxonomy. This was motivated by the fact that the "Model" figures had a skew above average, even though it was less than the skew for Graph figures.

- **Gabor filters.** Gabor filters are linear filters used for edge detection and texture analysis; it was used by Gkoufas et al. [23] to extract a total of 60 features.

Textual Features

- **Keywords.** Han and Chen [21] manually formed a 90-keyword feature vector of the most occurring words within captions of the figures in their training set. In such manner, each figure was represented using a binary vector of size 90, where an element would be set to 1 if the keyword related to its index existed in the figure caption, otherwise it

is kept at 0. Similarly, Kim et al. [22] formulated a vector of 568 keywords, but added an extra text pre-processing step prior to encoding the caption; the caption text was passed through a filter that automatically omitted numbers, special characters and stop words (e.g. almost, i.e., etc) and then applied Porter stemming to the remaining words.

- **Bag of words.** Bag of words is a common representation for document classification where the frequency of the word occurrence is taken into account rather than just its occurrence. Such representation was used by Rafkind et al. [20] alongside n-grams, for figure classification from the figure captions.

- **Character-based feature vector.** Following text extraction using an off-the-shelf tool, Ma et al. [19] used the extracted text to formulate a 37-dimensional vector to represent the following characters (A-Z, 0-9, Other). The effort by Ma et al. was specifically directed at identifying figures that contain gene sequences, and thus the frequency of A, C, G and T would make a big difference.

2.2.3 Autoencoders

Aside from feature engineering, automatic feature extraction methods such as autoencoders have been used extensively to automatically extract features from images, which are then used for classification [31–33]. Therefore, autoencoders could be used for the automatic extraction of visual features from the biomedical figure images, which is done in Chapter 4.

Autoencoders are feedforward neural networks designed to be trained in an unsupervised manner to reconstruct their inputs $\mathbf{x}$ into $\mathbf{r}$. Through this unsupervised training, autoencoders are able to learn some representation $\mathbf{h}$ of the input data $\mathbf{x}$, specifically when the representation is limited to a smaller size than the input. Autoencoders can be viewed as two components [34]:

1. The encoder function $f(\mathbf{x})$ that takes the inputs $\mathbf{x}$ and compresses them down into a specific code $\mathbf{h}$.

2. The decoder function $g(\mathbf{h})$, which takes that code layer $\mathbf{h}$ and tries to reconstruct it into $\mathbf{r}$. The decoder aims for $\mathbf{r}$ to be as similar to $\mathbf{x}$ as possible.

Specific restrictions are usually put on autoencoders to enable them to approximate inputs that resemble input data. Through such restrictions, the autoencoder has to adapt by prioritising specific aspects of the input, through which it learns some useful features from the input data. Autoencoders are trained to minimise the loss function $L(\mathbf{x}, g(f(\mathbf{x})))$ using training methods, such as SGD (Section 2.2.4).

Since the initial introduction of autoencoders, different variations have been put forward to serve specific tasks. Sparse autoencoders, for example, are aimed at classification tasks, where a sparsity penalty Ω is added to the training criterion for the code layer, thus turning the loss into the following:

$$L(\mathbf{x}, g(f(\mathbf{x}))) + \Omega(\mathbf{h}) \tag{2.1}$$

On the other hand, denoising autoencoders are more popular in computer vision tasks. They differ from traditional autoencoders by adding some type of noise to $\mathbf{x}$ and turning it into $\tilde{\mathbf{x}}$. The loss is then calculated using the following loss function:

$$L(\mathbf{x}, g(f(\tilde{\mathbf{x}}))) \tag{2.2}$$

2.2.4 Deep Learning

Over the past few decades, deep learning (DL) has gained huge momentum through advancements in statistics, applied mathematics and understanding of the human brain. More recently, developments in DL have accelerated not only due to its popularity and usefulness, but also due to the fast development of fast computers. DL feedforward neural networks approximate a function f^* through iteratively learning a parameter θ that leads to the best approximation of the following mapping $y = f(\mathbf{x}, \theta)$. Feedforward neural networks are composed of different functions that are chained together to form the most common structure of neural networks. For example, a network may have three functions f_1, f_2, f_3 connected in a chain that forms $f(x) = f_3(f_2(f_1(\mathbf{x})))$. Deep feedforward models are networks with an increasing depth of this chain [34], even though there is no agreed upon number of layers that defines the threshold between shallow and deep networks.

Prior the rise of DL, researchers sought hard-coded features (Section 2.2.2) that they believed would be helpful towards the classification or regression problem. However, as

the problems tackled by artificial intelligence become more complex, it simultaneously becomes more challenging to identify the features that should be extracted. This is the case for biomedical figure classification, considering the visual complexity of the figures as well as the subtle differences between some of the classes. One way to solve this issue is by taking the machine learning algorithm beyond simply mapping the features to the output space and into mapping raw inputs into representations. This has defined an entire research area called representation learning, which is concerned with methods to learn representations from input data to facilitate classification or other types of predictions.

Feedforward deep models are trained in a similar fashion to that of shallow neural networks using backpropagation to find the parameters θ. Stochastic gradient descent (SGD) is usually used, where given a set of inputs $\mathbf{x}_1, \ldots \mathbf{x}_N$, SGD aims to minimise:

$$\theta = \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(\mathbf{x}_i, \theta) \tag{2.3}$$

where L is the selected criterion to calculate the learning loss. Such training is done using mini-batches, where each mini-batch contains a subset of the training data $\mathbf{x}_1, \ldots, \mathbf{x}_m$. The gradients are then calculated using:

$$\frac{1}{m} \frac{\partial L(\mathbf{x}_i, \theta)}{\partial \theta} \tag{2.4}$$

where m is the mini-batch size, which is pre-selected depending on the dataset with the intention that m should be large enough to provide a good representation of the data.

Residual Networks

One of the biggest challenges of going deeper with neural networks is error degradation. This was reflected in the work done by Simonyan and Zisserman [35], in which they noticed a saturation level once their deep architecture has more than 19 layers. He et al. [36] proposed a skip connection that is employed as a building block for deeper architectures, which can overcome training degradation. The residual building block is formally defined as:

$$\mathbf{y} = F(\mathbf{x}, \mathbf{W}_i) + \mathbf{x} \tag{2.5}$$

where $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{W}_i$ are the block input, output vectors and weights respectively, and $F + \mathbf{x}$ is performed using the skip connection and element-wise addition. Residual networks (ResNet) have prevailed in deep architectures because of their ability to reduce training degradation without adding more parameters to the model and without adding any computational complexity.

Deep Learning for Classification

DL models have prevailed in the last decade with classification problems, especially in computer vision. This has been largely due to the growth in computational power, allowing for much larger and more capable models. This has allowed DL models to catch up with the increasing sizes of datasets and learn the necessary representations to differentiate between an increasing number of classes. Modern deep networks have built the ability to recognise $1,000$ different classes, as illustrated by contributions made towards one of the largest object recognition contest, ImageNet [37]. Similarly, DL can be used for the classification of biomedical figures, where the number of classes is comparatively large along with a large number of examples from biomedical articles.

In classification, DL feedforward neural networks aim to learn $y = f^*(x)$ for a set number of classes. The number of classes reflects the size of the output layer of the feedforward network. To make $f(x)$ better at predicting y, the outputs of the network are compared at each iteration with the ground truths using a specific criterion. Cross-entropy is the criterion that is used to calculate the error between the outputs of the model and the target outputs. Cross-entropy can be closely correlated to Kullback-Leibler (KL) divergence, which is considered as a method to measure what is conceptualised as a distance between two probability distributions. More formally, cross-entropy is defined as follows:

$$C = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}\left(y_i^{(j)}\ln(o_i^{(j)}) + (1-y_i^{(j)})\ln(1-o_i^{(j)})\right) \tag{2.6}$$

where N is the total number of samples in the training set, C is the number of classes (also the number of output neurons), while y_i and o_i are the target and outputs for sample i [38].

Deep Learning for Biomedical Figure Classification

In a move away from hand-crafted features, DL models were developed for the classification of biomedical figures. Koitka and Friedrich [39] demonstrated how DL methods can outperform other machine learning models that use hand-crafted features. More specifically, a ResNet-152 model was able to surpass all other methods based on visual features from the figures. Following that, Kumar et al. [26] used ensembles of different popular DL architectures, AlexNet [40] and GoogLeNet [41], to classify biomedical figures. A novel approach was then employed by Zhang et al. [25] using a dual ResNet approach, where two figures are input at the same time and the overall model would make a decision on whether the figures belong to the same class or not.

2.2.5 Support Vector Machines

Support vector machine (SVM) is a powerful classification method that has been widely used for many classification tasks in general and for biomedical figure classification in particular [18, 24, 39, 42, 43]. This section describes the theoretical background behind multi-class SVMs, which are used to classify biomedical figures in Chapter 4.

Starting from labelled training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. Assuming that the two classes are separable in $\mathbb{R}^d$, the following inequality will be satisfied for a vector $\mathbf{w}$ and scalar β:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + \beta) \geq 1 \quad \forall i \tag{2.7}$$

Support vector classifiers [44] aim to find the hyperplane $f(\mathbf{x})$ that can separate the data in a way that creates the largest possible margin M between class -1 and 1:

$$f(\mathbf{x}) = \mathbf{w}_0\mathbf{x} + \beta_0 = 0 \tag{2.8}$$

This defines $\frac{\mathbf{w}}{|\mathbf{w}|}$, which has the maximal distance between the projections of the training vectors of the two classes. The classification rule for a sample $\mathbf{x}_i$ then becomes dependent on

which side of the hyperplane the sample falls on, which in this case can be defined as:

$$g(\mathbf{x}_i) = sign(f(\mathbf{x}_i)) \tag{2.9}$$

The optimal margin M can be defined as the distance between the minimum projection of class 1 onto $\frac{\mathbf{w}}{|\mathbf{w}|}$ take away the maximum projection of class -1 also onto $\frac{\mathbf{w}}{|\mathbf{w}|}$. M can then be expressed as:

$$M = \min_{x:y=1} \frac{\mathbf{x} \cdot \mathbf{w}}{|\mathbf{w}|} - \max_{x:y=-1} \frac{\mathbf{x} \cdot \mathbf{w}}{|\mathbf{w}|} \tag{2.10}$$

Therefore, finding the best hyperplane becomes about maximising M, thus the optimal hyperplane is one that minimises $\mathbf{w}.\mathbf{w}$. This makes the task of finding an optimal hyperplane a quadratic programming problem, which is simplified by identifying the so called support vectors $\mathbf{x}_i$, where $y_i(\mathbf{w}.\mathbf{x}_i + \beta) = 1$.

However, in most datasets, the classes overlap in the feature space, making it imperative to accept some level of compromise when determining the separating hyperplane. The support vector classifier in this case still aims for what is called a soft-margin, where some slack variables $\xi = (\xi_1, \ldots, \xi_N)$ are permitted to be within the margin and thus the initial constraint in Equation 2.7 becomes:

$$y_i(\mathbf{w}.\mathbf{x}_i + \beta) \geq 1 - \xi_i \quad \forall i \tag{2.11}$$

where $\forall i$, $\xi_i \geq 0$, $\sum_{j=1}^{N} \xi_j \leq$ constant. Minimising the sum errors $\sum_{i=1}^{N} \xi_i$, can lead to a minimal subset of training error, which when can be excluded from the overall training set. The remaining subset can be separated using an optimal hyperplane as previously done, which can be formalised through:

$$\min_{\mathbf{w},\beta,\xi} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l} \xi_i \tag{2.12}$$

where $C > 0$ is the regularisation parameter.

Many approaches exist to extend SVM classifiers from binary to multi-class classifiers, such as all-together, one-against-all and one-against one. LIBSVM [45] implements a one-against-one approach, which was proved as a competitive approach by Hsu and Lin [46] following a comprehensive comparison between the different approaches towards multi-

class SVMs. Given k number of classes, the one-against-one approach requires a total of $k(k-1)/2$ classifiers, where each one is fitted using the training data of two classes. For this multi-class problem, finding the soft margin in Equation 2.12 becomes:

$$\min_{\mathbf{w}^{ij},\beta^{ij},\xi^{ij}} \quad \frac{1}{2}(\mathbf{w}^{ij})^T\mathbf{w}^{ij} + C\sum_{z=1}^{l}(\xi^{ij})_z \tag{2.13}$$

SVMs have been widely used for many classification and regression problems in various areas. Starting with bioinformatics, where SVMs have been used for cancer classification from DNA micro-arrays [47], all the way to hand writing recognition [48] and face detection [49].

Even though SVMs can be defined as a two-layer network [44], DL has gained momentum ahead of SVMs in recent years due to many advancements that have allowed for models to learn deeper representations.

2.2.6 Linear Discriminant Analysis

Linear discriminant analysis (or Fisher linear discriminant analysis) [50] is a fundamental tool of multivariate statistics used in machine learning and pattern recognition. It is used to linearly classify a number of classes, or more commonly, reduce the dimensionality of the feature space for further classification. Unlike support vector classifiers, the LDA decision boundary is calculated through the covariance of the class distributions and the positions of the class centroids. LDA is closely related to principal component analysis, as both methods are based on linear transformations. However, unlike PCA, LDAs take the classification of the points into account, while reducing the dimensionality of the data. The focus of the transformation process with LDA is to maximise the ratio of the between-class variance over the inner-class variance.

Starting from a labelled set of data $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_N,y_N)\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{C_1,\ldots,C_c\}$. Multi-class LDA aims to find the projection matrix $\mathbf{\Theta}^*$ that maximises the ratio $J(W)$ between between-class scatter and within-class scatter:

$$J(W) = \frac{|\tilde{\mathbf{S}}_B|}{|\tilde{\mathbf{S}}_W|} = \frac{\mathbf{\Theta}^T\mathbf{S}_B\Theta}{\mathbf{\Theta}^T\mathbf{S}_W\Theta} \tag{2.14}$$

where:

$$\tilde{\mathbf{S}}_B = \sum_{i=1}^{|C|} N_i(\hat{u}_i - \hat{u})(\hat{u}_i - \hat{u})^T \tag{2.15}$$

$$\tilde{\mathbf{S}}_W = \sum_{i=1}^{|C|} \sum_{y \in C_i} (y - \hat{u})(y - \hat{u})^T \tag{2.16}$$

and:

$$\hat{u}_i = \frac{1}{N_i} \sum_{i \in C_i} y_i \tag{2.17}$$

$$\hat{u} = \frac{1}{N} \sum_{i}^{N} y_i \tag{2.18}$$

There has been a recent focus on exploiting the powerful aspects of LDA to learn linearly separable features using deep neural networks [51, 52]. Stuhlsatz et al. [52] used a linear discriminant criterion to fine-tune a pre-trained stack of restricted Boltzmann machines (RBMs). Dorfer et al. [51] on the other hand, focused on the entire training procedure of deep learning models, where they switched the focus of the training process from maximising the likelihood of target classes to an output of feature distribution that adheres to the LDA objectives. The objective function was therefore derived from the LDA eigenvalue problem and was designed to function with SGD and back propagation.

2.2.7 Classification and Data Imbalance

Class imbalance is a problem that occurs when one class is insufficiently represented in the dataset. This generally leads classification models to be biased towards the more represented classes.

SVM tries to deal with this challenge using different penalty parameters in Equation 2.12 for the different classes in the dataset [53], thus transforming the problem into:

$$\min_{\mathbf{w},\beta,\xi} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C^+ \sum_{y_i=1} \xi_i + C^- \sum_{y_i=-1} \xi_i \tag{2.19}$$

where C^+ and C^- are both greater than 0 and used to regularise the cost for the positive and negative classes respectively.

The cross-entropy criterion deals with the data imbalance in a similar manner, where the error is scaled differently between the classes. The cross-entropy in Equation 2.6 becomes:

$$C = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} w_j\left(y_i^{(j)}\ln(o_i^{(j)}) + (1-y_i^{(j)})\ln(1-o_i^{(j)})\right) \tag{2.20}$$

where w_j is the weight for class j, which is used to rescale the loss for that class according to the intended impact.

As for LDAs, no negative effects could be empirically linked with class imbalance [54]. Xue and Titterington [54] had to make this argument against a previous claim by Xie and Qiu [55], who claimed a theoretical negative effect of unbalanced data on LDA performance. Xue and Titterington [54] used a more reliable metric to assess the LDA performance empirically, where they used the misclassification error rate rather than the Area Under the receiver operating characteristic Curve (AUC) used by Xie and Qiu [55].

2.3 Text Extraction

Extracting text from digital images is an important task that has been extensively researched, whether from scanned documents or from natural scene images. The research that has been done involves the different stages of text extraction, starting with text localisation and ending with text correction. Text extraction challenges have been established from natural scene images [56] and complex documents [57]. Recently, extracting text from biomedical figures has gained attention and a challenge specific to text extraction from biomedical figures was initiated using the DeTEXT dataset [58]. Extracting text from biomedical figures is a unique task because of the nature of the images and the nature of text contained within them.

2.3.1 Text Extraction from Biomedical Figures

The different approaches towards extracting text from biomedical figure images have mostly focused on the image pre-processing to enhance the output quality of off-the-shelf OCR tools. This section compares many of the different approaches that investigated text extraction

from biomedical figures or tried to utilise the text contained with the figures for further applications.

Xu et al. [13] identifies elements of text from biomedical figures and then employs OCR techniques to extract their text. The extraction process is carried out in two stages to improve the extraction accuracy, one that employs a cross-checking procedure (aimed at higher precision) and another that does not (aimed at higher recall). The cross-checking procedure involves checking whether the extracted word exists in the article's body of text, including its figure captions. If the extracted word does not occur in the body of text, the word is then discarded. However, this step is counter-productive for the purpose of indexing figures using the text contained within them, because articles are already indexed using words contained within them. Targeting vertical texts contained within biomedical figures, the images are also rotated 90° before applying OCR again. The extracted text was incorporated as part of their indexing for the Yale Image Finder search engine [13].

Xu and Krauthammer [59] proposed an iterative pivoting text region detection method that constructs vertical and horizental histograms to locate text regions. After the iterative process is done with generating candidate text regions, they assess each region using an overall edge density heuristic that removes regions that have a density that is above or below specific thresholds. The iterative nature of their approach was targeted at dealing with the distributed nature of text in biomedical figures. Xu and Krauthammer [60] then employed their method with off-the-shelf OCR engines to extract text from biomedical figures, comparing the extraction results before and after localisation. They used two OCR engines, the first was the Microsoft Document Imaging package (from the Microsoft Office 2003 suite). The second was Top OCR[1], which is a free OCR engine.

Kim et al. [61] proposed a figure text extraction tool (FigTExT) to improve the ability of OCR tools to extract text from biomedical figures. The proposed tool focuses on text localisation, image pre-processing and text correction to enhance the text extraction accuracy. Firstly, Kim et al. employed methods used to localise text in natural scene images to localise text in biomedical figure images. More specifically, they employed Gatos's et al. approach [62] that is based on connected component analysis. This was repeated for the image and its inverse to cope with an unknown grey-level for text regions in the figures. Kim et al. argued

[1]https://www.topocr.com/topocr.html

that Gatos's et al. method performs well with high contrast text regions such as the ones in biomedical figures. The extracted text regions were then up-sampled to improve upon the image quality, where a contrast stretching transformation was done to improve the contrast of the image. Following pre-processing, the text regions were input into the SimpleOCR API[2] to recognise the specific characters.

In their proposed framework towards image-based document retrieval, Lopez et al. [11], employed ABBYY Finereader[3], a commercial OCR software, to localise and extract text from biomedical figures. The same software was also used by Ma et al. [19] to extract text towards the classification of biomedical figures. The extracted text was then used to classify whether a figure contains gene sequences based on the occurrence of the A, C, G and T characters (see Section 2.2.2). Considering that their ultimate goal was to classify figures, the text extraction performance was not of very high value compared to other applications for the text, such as indexing.

2.3.2 Deep Learning for Text localisation

Text detection has been approached in various ways, some methods are sliding window based [63–65], where the aim is to restrict the search for text to a limited subset of regions. Each candidate region from the subset is then input into a classification algorithm that determines whether the region contains text or not. On the other hand, some text detection methods are based on connected components [66–68], which search for text within an image through grouping pixels with similar properties together into a single text region. Furthermore, there exist hybrid methods that implement connected component analysis alongside sliding window methods [69, 70].

More recently, and similarly to other areas of computer vision, text detection and localisation has witnessed rapid advances because of the rapid developments in deep learning. More specifically, deep convolutional neural networks (DCNNs) have been used as a classification mechanism to identify regions of text from the candidate subset proposed by a sliding window [71]. This is conceptually similar to current object detection methods, where DCNNs depend on region proposals or sliding windows to find objects [72]. However, the YOLO detection

[2]https://www.simpleocr.com/
[3]https://www.abbyy.com/en-gb/finereader/

system [73] redefined the object detection problem as a single regression problem, thus reducing the complexity of the training process.

2.4 Summary

The chapter provided a look into the two big themes of this thesis, biomedical figure classification and text extraction. Starting by defining the biomedical figure classification task and the taxonomy of biomedical figures. Then moved on to survey the different hand-engineered visual and textual features that have been used for the classification of biomedical figures. This survey made it abundantly clear that it is important to develop methods to automatically extract features from biomedical figures. Therefore, the chapter then proceeded to automatic feature extraction methods such as autoencoders, which were the essence of the feature extraction process done in Chapter 4 as well as being the inspiration behind the training procedure proposed in Chapter 5 for text localisation. Continuing on with automatic feature extraction, deep learning was discussed with a focus on classification, including the effects on computer vision and the classification of biomedical figures. It was evident that advancing deep learning approaches can contribute to further enhance the classification of biomedical figures.

The chapter also covers two important classification methods, SVM and LDA. It is shown how SVMs have been used in bioinformatics in general and for biomedical figure classification in particular. However, SVMs require improvements and adjustments to make them more effective for the classification of figures in biomedical literature, especially when considering the hierarchical nature of the biomedical figure taxonomy. Data imbalance was discussed with both SVMs and LDAs and it was argued that LDAs are less impacted by class imbalance. Therefore, LDAs are used as the base of the methods presented in Chapter 3.

Finally, the chapter looked into text extraction from biomedical figures and the different approaches that tried to address the challenge. There is a clear lack of deep learning approaches into the extraction of text from biomedical figures, which is addressed in Chapter 5.

Chapter 3

Effective Representation Learning

"Generally speaking, a good representation is one that makes a subsequent learning task easier."

Ian Goodfellow
Yoshua Bengio
Aaron Courville

3.1 Introduction

Effective representation learning is a key factor towards the success of machine learning algorithms such as: clustering and classification. In such manner, learnt representations can make the task at hand either very easy or very difficult depending on the way the inputs are represented. Thus, representations that are to be learned should be dependent on nature of the learning task at hand. This chapter offers a fresh look into the supervised training of deep neural networks for classification. Specifically proposing a new method to train a deep neural network towards the gradual separation of features in a new reduced space. Keeping with the theme of this thesis of extracting information from biomedical figures, this chapter is aimed at improving classification performance for deep learning models. Thus, prior to implementing this method for biomedical figure classification, this chapter presents the theoretical background for the proposed method. More specifically, this method is later

employed in Chapter 4 to train deep learning models for the classification of biomedical figures.

The proposed method, named GCS (Gradual Class Separation), is described in this chapter; starting with a mathematical formulation of the classification task and establishing its link with Fisher's linear discriminant analysis. After that, the chapter discusses how GCS functions with deep neural networks, the mean squared error and support vector machines towards classification. The chapter then briefly describes the experimental settings used to compare the performance of GCS with other established methods. The experimental settings include details of the deep neural network architectures used as well as the benchmark datasets (CIFAR-10 and CIFAR-100). Results from our GCS method are then compared to those from similar models trained using the cross-entropy criterion. The results are then analysed with a special regard to the method's generalisation ability and its computational complexity.

3.2 Feature Reduction and Maximization of Inter-Class Distance

Starting from the following empirical data:

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$$

where $\mathbf{x}_i \in \mathbb{R}^D$ is an input vector that has a single label $y_i \in \{C_1, C_2, ..., C_k\}$, where k is the total number of classes, and each class C_j has N_j number of samples.

The aim is to train a neural network to learn a transformation function $f(\mathbf{x}_i) = \mathbf{x}'_i$, where $\mathbf{x}'_i \in \mathbb{R}^d$ and $d \ll D$. Therefore, the new reduced data will be:

$$(\mathbf{x}'_1, y_1), (\mathbf{x}'_2, y_2), \ldots, (\mathbf{x}'_n, y_n)$$

In this reduced space, the neural network has to also maximise the inter-class distance (also referred to as between-class scatter), which can be accomplished by increasing the distances

between the mean vectors:

$$d(C_1, C_2, \ldots, C_k) = d(\mathbf{m}'_1, \mathbf{m}'_2, \ldots, \mathbf{m}'_k) \tag{3.1}$$

where:

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{i=1}^{N_j} \mathbf{x}_i \quad \text{where} \quad y_i \in C_j \tag{3.2}$$

$$\mathbf{m}'_j = \frac{1}{N_j} \sum_{i=1}^{N_j} \mathbf{x}'_i \quad \text{where} \quad y_i \in C_j \tag{3.3}$$

where N_j is the total number of samples from class C_j. However, maximising the distance between classes is not sufficient, and therefore while maximising the inter-class distance, the inner-class distances have to be minimised [74]. This is similar to the objective behind Fisher's linear discriminant analysis (LDA) [75], which aims to find $\mathbf{\Theta}$ that maximises the ratio $J(w)$:

$$J(w) = \frac{\mathbf{\Theta}' S_B \mathbf{\Theta}}{\mathbf{\Theta}' S_W \mathbf{\Theta}} \tag{3.4}$$

where:

$$S_W = \sum_{i=1}^{k} \sum_{j=1}^{N_i} (\mathbf{m}_j - \mathbf{m}_i)(\mathbf{m}'_j - \mathbf{m}'_i) \tag{3.5}$$

$$S_B = \sum_{i=1}^{k} N_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}'_i - \mathbf{m}') \tag{3.6}$$

where:

$$\mathbf{m} = \frac{1}{k} \sum_{i=1}^{k} \mathbf{m}_i \tag{3.7}$$

$$\mathbf{m}' = \frac{1}{k} \sum_{i=1}^{k} \mathbf{m}'_i \tag{3.8}$$

and $\mathbf{m}_i$ is the class mean given by Equation 3.3, but in the original space. From the objective function $J(w)$ given in Equation 3.4, LDA searches for the projection $\mathbf{\Theta}$, that maximises the ratio between the inter-class scatter and inner-class scatter. However, in this chapter, the aim is to get a neural network to learn the transformation that maximises the ratio between the inter-class scatter and inner-class scatter in the reduced space $\mathbb{R}^d$.

Supervised training of feedforward neural networks naturally means that different levels of representations of the inputs will be learnt at every hidden layer and more so as we go

deeper into the model. Those representations are typically learnt from the error calculated at the linear classifier, which usually forms the last layer of the network, and is propagated back through the model. In this work, the last linear classification layer is removed and the error calculation is moved to the model's penultimate layer, which is expected to obtain linearly separable features.

Neural networks are normally trained towards classification using the cross-entropy criterion that depends on its two components: the negative log likelihood (NLL) and a log Softmax function as follows:

$$NLL(\mathbf{o}_i) = -\log(\mathbf{o}_i) \tag{3.9}$$

$$LogSoftMax(\mathbf{o}_i) = \log\left(\frac{e^{\mathbf{o}_i}}{\sum_j e^{\mathbf{o}_i}}\right) \tag{3.10}$$

$$CrossEntropy(\mathbf{o}, c) = -\mathbf{o}_c + \log\left(\sum_{i=1}^{len(\mathbf{o})} e^{\mathbf{o}_i}\right) \tag{3.11}$$

where $\mathbf{o}$ is the output vector and c is the target class. Following this, the error is propagated back through the neural network layers using back propagation. As a consequence of this, the hidden layers start learning varying levels of features that can differentiate between the different classes. However, in this work, a question is asked: what if a neural network can be directly trained to separate features in a latent space, while maximising the inter-class distance and minimising the inner-class distance? A gradual separation of features is proposed by continuously moving ground truths, which aims to slowly move the classes apart from each other in the latent space. Moving ground truths is dependent on the labels, making this method a supervised training method.

Consider a batch $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_b, y_b)$ from $\mathbb{R}^D$ and its reduced counterpart $(\mathbf{x}'_1, y_1), \ldots, (\mathbf{x}'_b, y_b)$ in $\mathbb{R}^d$, where $b \geq 1$. The features $(\mathbf{x}'_1, \ldots, \mathbf{x}'_b)$ are mapped out in the reduced space $\mathbb{R}^d$ using their labels $(y_1, \ldots, y_b)$, and the batch mean vector $\mathbf{m}'_c$ is calculated for every class c using Equation 3.3. The overall mean $\mathbf{m}'$ is also calculated for the batch using Equation 3.8. New target mean vectors $\mathbf{m}''_c$ are calculated for each class and for every batch:

$$\boldsymbol{m}''_c = \mathbf{m}'_c + \boldsymbol{\Delta}_c \tag{3.12}$$

where:

$$\boldsymbol{\Delta}_c = [\delta_1, \ldots, \delta_d] \tag{3.13}$$

where:

$$\delta_j = \lambda \sum_{i=1}^{k} v\left(\mathbf{m}_c'^{(j)} - \mathbf{m}_i'^{(j)}\right) \quad for \quad j = 1, \ldots, d \tag{3.14}$$

where v is a function that sends a vector component x to $|x|^{-1/2} \cdot \text{sgn}(x)$, and λ is a regularisation parameter.

$\boldsymbol{\Delta}_c$ moves each class's batch mean vector $\mathbf{m}'_c$ in the opposite direction of the other classes in every dimension of $\mathbb{R}^d$, and $\text{sgn}(x)$ is used to preserve the movement direction, especially after taking the absolute value of the different elements $(\mathbf{m}_c'^{(j)} - \mathbf{m}_i'^{(j)})$ that enables the use of square root. Through v, it is possible to apply a non-linear relationship between the distances between two classes and the resulting movement, which makes it possible to turn smaller distances into larger displacements.

Every input $\mathbf{x}_i$ from $\mathbb{R}^d$ that belongs to class c is assigned the same ground truth in that batch, and the error is calculated using the following mean squared error (MSE):

$$MSE(\mathbf{x}'_i, c) = (\mathbf{x}'_i - \mathbf{m}''_c)^2 \tag{3.15}$$

As $\mathbf{m}''_c$ is set to be the ground truth for every $\mathbf{x}_i$ from the same class c (Equation 3.12), the aim becomes about more than just moving the classes apart in the feature space, by also focusing on reducing the inner-class sparsity.

3.3 Classification in the Reduced Space

Following on from the feature extraction process, the focus turns towards using the extracted features towards classification. Described in Section 2.2.5, an SVM is a powerful classification tool that uses non-linear mapping to map input features into a new high-dimensional feature space. In this new feature space, the SVM seeks to find the hyperplane that provides the maximum margin between the classes. Like other classification methods, SVMs seek the best separation from the training samples, whilst maintaining the model's generalisation capabilities. The classification accuracy and computational efficiency for SVMs is also

impacted by irrelevant features, such as redundant features, outliers and noise. Generally in classification, reducing the dimensionality of the input feature space overcomes the risk of overfitting on the training data, and thus enhances the chances for better generalisation. Overfitting becomes a big problem when the number of training samples is comparatively small compared to the number of features [47].

In this work, instead of training a Softmax classifier, an SVM is employed to classify the outputs of the last layer. In this manner, the classification power of the SVM is harnessed, while the error is propagated through the model using the proposed method that aims to maximise the distances between the classes and reduce the inner-class sparsity for each class. This goal indirectly leads to a larger margin between the classes in the latent feature space. This is why the choice was made to use an SVM for classification, where the separation goal of GCS meets with the margin maximisation goal of the SVM.

To further optimise the performance, and because the training is not dependent on fitting the SVM, the classification is only done for validation after every specific number of iterations. This approach is motivated by research that has shown significant effect of feature selection on the SVM classification accuracy and computational efficiency [47, 76].

Hinge loss is a similar approach to GCS, which is used to find the soft-margin for SVMs and is also used to train neural networks (see Section 2.2.5). Multi-label margin loss (also known as margin-based loss) is a similar function that is used to train neural networks towards classification, where the loss is calculated as follows, for input vector $\mathbf{x}$ and target $\mathbf{y}$:

$$loss(\mathbf{x}, y) = \sum_{ij} \frac{\max(0, 1 - (\mathbf{x}_y - \mathbf{x}_i))}{len(\mathbf{x})} \tag{3.16}$$

Similarly, Elsayed et al. [77] proposed a loss function that depends on the first-order approximation of a margin to separate classes. Their proposed method is similar to ours in the manner that it can be used to calculate errors at various depths of the model. However, even though their proposed method focuses on separating classes in the latent feature space, their method does not account for within-class spread.

Parallelism

Improving the classification timing, instead of using a single SVM, an ensemble of SVMs is fitted on the reduced features of the input data. In this way, it was possible to fit each SVM from the ensemble on a different CPU core. This is achieved by splitting the total number of input samples equally among each SVM in the ensemble.

3.4 Experimental Settings

3.4.1 Datasets

Two datasets with varying number of classes have been selected to assess the effectiveness of the proposed learning method. Starting with a relatively small number of classes with CIFAR-10 (10 classes) and ending with a large number of classes with CIFAR-100 (100 classes). Both datasets have perfect balance between their classes, so in this chapter we focus on establishing the GCS method with (class) balanced datasets. Whereas, in the next chapter, the focus will be on dealing with class imbalanced data.

CIFAR-10

CIFAR-10 [78] is a dataset that contains $60,000$ small coloured images ($3 \times 32 \times 32$) split evenly into 10 classes, with $6,000$ in each class. Each class in the dataset has $5,000$ samples for training and another $1,000$ for testing. The classes in the dataset: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. The classes of CIFAR-10 are entirely mutually exclusive, where none of the classes overlaps with another. CIFAR-10 is a popular benchmarking dataset due to the nature of its classes and their even distribution. In this chapter, the same data split for this dataset has been applied throughout when testing the different methods.

CIFAR-100

A similar dataset to CIFAR-10, but with 100 classes instead of 10, where each class contains 600 samples. The 600 samples for each class are split into 500 training images and 100 for testing. The 100 classes of the CIFAR-100 dataset belong to 20 different super-classes, giving its taxonomy a hierarchical structure. Each sample is made out of a 32×32 sized RGB image with two labels, one for the class and another for the superclass. The total number of images in the CIFAR-100 dataset is the same as that for CIFAR-10, thus making the dataset aimed at assessing a model's prediction capability with a large number of classes and a much lower number of samples per class.

3.4.2 Models

To test with a number of benchmark deep models with varying depths, deep residual networks and VGG models were used. The choice was first made to work with the benchmark deep residual models, but to ensure that impact of GCS was not limited to models with the skip layers, VGG models were also considered.

Deep Residual Neural Networks

Deep residual neural networks (ResNets) [36] are feedforward neural networks that use skip connections to address the degradation problem in deep models. ResNet models were able to push the boundaries of deep learning using their skip connections that allow for better error propagation through the model. ResNets utilise two main types of building blocks for their network architecture: the first is the standard residual building block and the other is "bottleneck" building block, which is used for deeper models. The different ResNet variations can all be divided into three main components:

1. The input convolution followed by a max-pooling layer.
2. Four blocks that can be either standard residual blocks or bottleneck blocks, depending on the depth of the model.

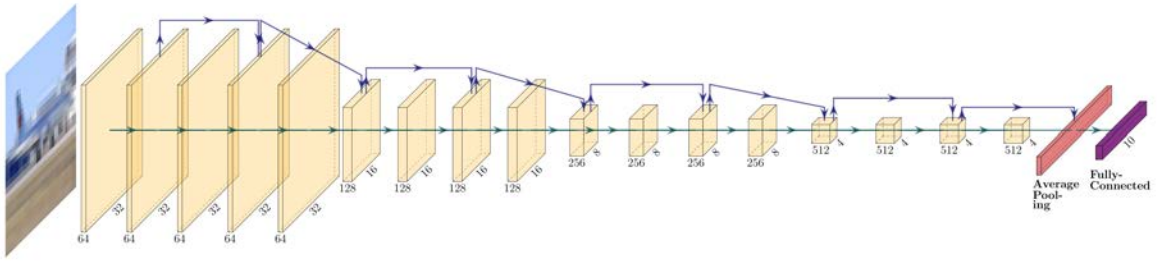


Fig. 3.1 ResNet-18 architecture.

3. An average pooling layer connected to fully-connected layer that has a number of neurons that matches the number of classes to be predicted followed by a Softmax activation function. The number of outputs from the average pooling layer is dependent of the type of the block used in the second component, if the basic block is used then there would be 512 features while there would be 2048 features if the bottleneck block was used.

The Resnet-18 architecture is shown in Figure 3.1, where those different components are visible. For our proposed method, only the last component had to be altered, where the Softmax function was replaced by ReLU and a 1-dimensional batch normalisation layer. The number of neurons in the fully-connected layer was also changed to match the number of features rather than the number of classes.

VGG

The VGG model [35] is another form of deep feedforward neural network that was around before the ResNet model. The VGG model made the improvement over AlexNet [40] by replacing large kernel-sized filters with multiple 3×3 filters one after the other. However, unlike ResNet, the VGG model does not use the skip layers and is therefore more prone to the degradation problem. In a similar manner to ResNet models, the VGG model also has variations depending on the depth of the model. The VGG model architecture also has three main components:

1. The input layer, which is dependent on the input dimensions.

2. A stack of convolutional layers with different depth depending on the architecture.

3. Three fully connected layers, with the first two having 4096 neurons and the last one having a number of neurons that matched the number of classes in the dataset. The final layer incorporates a Softmax activation function that is responsible for outputting the probability for each input belonging to a specific class.

In a similar manner to the ResNet architecture, the VGG architecture was altered slightly by replacing the Softmax layer with a ReLU function and adding a batch normalisation layer after that. The number of neurons in the last layer was also set to the intended number of features rather than the number of classes. The VGG-19 was used to provide an architecture of a similar depth to the ResNet-18.

3.4.3 Training

Similar depths of the ResNet and VGG models were trained to ensure the functionality of GCS with and without the use of skip layers in the architecture. Each model was first trained using the cross-entropy criterion and then using our proposed method to compare the difference in performance and efficiency. Each of the models was first trained to minimise the cross-entropy loss (L) for each sample output (o):

$$L(\mathbf{o}, c) = -\mathbf{o}_c + \log\Big(\sum_{j}^{len(\mathbf{o})} e^{\mathbf{o}_j}\Big) \tag{3.17}$$

where c is the target class. This criterion is a combination of the negative log likelihood and log Softmax functions (see Section 3.2). Whereas the loss according to our GCS method is based on the mean squared error, where the error is calculated between the outputs and the new mean vector for the class:

$$L(\mathbf{o}, c) = (\mathbf{o} - \mathbf{m}''_c)^2 \tag{3.18}$$

where $\mathbf{m}''_c$ is calculated from Equation 3.12. The value for λ was set to a small value at the start of the GCS experiments, and then through the iterations it was increased exponentially, setting its value at iteration i to:

$$\lambda_i = \lambda^i - 1 \tag{3.19}$$

The increase of λ aims to increase the impact of smaller distances between classes, and therefore lead to higher error. The exponential growth was to counter the slow-down in learning that occurs when using MSE.

The batch loss was then calculated by taking the average loss (L) over all the batch observations. The same batch size of 120 was used when training the models with the cross-entropy loss and with the proposed method. The error was propagated back through the model after every batch, while using two different optimisation algorithms. The first algorithm used with the cross-entropy criterion was stochastic gradient descent (SGD) with a momentum of 0.9 and a weight decay of 5×10^{-4}. The initial learning rate was set to 0.1 with learning milestones set to iterations 150, 250 and 350, where the new learning rate was calculated $lr_i = lr_{(i-1)} \times 0.1$ for iteration i. While, the ADADELTA [79] optimisation algorithm was used for our proposed method, because of its ability to dynamically adapt during training solely using first order information. Additionally, ADADELTA is robust to noisy gradient information, which is very suitable for our evolving ground truths. ADADELTA also does not require the manual setting of a learning rate or any type of decay, as it is adaptive.

The SVM was fitted gradually using the outputs of each batch from the validation iteration. After comparison between the performances of different kernel functions for the SVM, the linear function prevailed over both polynomial and radial basis. This was also part of the aim, which is to check the level of linear separability that has been reached. The cost value for the SVM was set to 10 and then it was increased with the same ratio as the class separation, which is discussed in more detail in Section 3.5.1. The maximum number of iterations was set to $100,000$ to avoid stalling on the SVM part during fitting, while the tolerance for stopping criterion was set to 10^{-3}. To save on training time, Shrinking technique [80] was also employed to eliminate certain points that are unlikely to turn out to be support vectors.

In addition to the batch normalisation layer, which as the name suggests operates on batch level, the output of each batch was used to standardise the features by removing the mean and scaling according to the unit variance:

$$x' = \frac{(x - \mu)}{\sigma} \tag{3.20}$$

where μ and σ are the mean and the standard deviation respectively, calculated from the training samples. μ and σ are calculated from every training batch of the validation iteration and the values are then used to transform the features from the test samples.

Other than the SVM, a two layer fully-connected neural network was also used for classification along with GCS. The input size of the classification network matches the number of output features, while its output Softmax layer has the same number of neurons as the number of classes. The output features from each batch were input into the classification model, which in turn was trained using cross-entropy. ADADELTA was also used as an optimiser for this model with an initial learning rate set to 1.

Choosing the optimal number of dimensions was another challenge, which was tackled through experimenting with different number of features. Starting with the number of features that was closer to the number of features used by the last feature layer of the ResNet-18 and ResNet-34 architectures, which is 512. In this chapter, the number of features was always set to 500, whether the model used was the ResNet-18 or VGG-19. It is worth noting that experiments showed that it is possible to use a smaller number of features to achieve a similar performance. However, it was decided that finding an "optimal number of features" for each of the tackled tasks would be out of the thesis.

Image Pre-processing

The means ($\mathbf{m}^{(c)}$) and standard deviations ($\mathbf{std}^{(c)}$) are calculated for each channel (c) of the CIFAR datasets from their training sets and the input values for each channel ($\mathbf{inp}^{(c)}$) are normalised according to:

$$\mathbf{inp}^{(c)} = \frac{\mathbf{inp}^{(c)} - \mathbf{m}^{(c)}}{\mathbf{std}^{(c)}} \tag{3.21}$$

The images were also randomly cropped to 32×32 after they were padded by 4 from every side.

3.5 Evaluation

Because the datasets used in this chapter are multi-class datasets, the average accuracy was calculated for all the experiments carried out:

$$\text{Average Accuracy} = \frac{1}{|C|}\sum_{i=1}^{|C|}\frac{TP_i + TN_i}{TP_i + TN_i + FP_i + TN_i} \tag{3.22}$$

where:

- TP_i is the total number of samples correctly classified as class i.
- TN_i is the total number of samples correctly classified as not belonging to class i.
- FP_i is the total number of samples falsely classified as class i.
- FN_i is the total number of samples falsely classified as not belonging to class i.
- C is the set of classes in the dataset.

3.5.1 Fisher's Ratio

It is important here to remember the main aim of the proposed method, which is to separate the classes in the latent feature space while also minimising the inner-class scatter. Therefore, to evaluate the level of class separation achieved, firstly the following static metric is defined:

$$E = \frac{\|S_B\|}{\|S_W\|} \tag{3.23}$$

with S_B and S_W being the between and within class scatter matrices, respectively.

$$E^{(i)} = \frac{\|S_B^{(i)}\|/\|S_W^{(i)}\|}{\|S_B^{(1)}\|/\|S_W^{(1)}\|} \tag{3.24}$$

where $E^{(i)}$ is a value that shows how the class separation at iteration i compares with that of iteration 1. At each iteration the separation is judged compared to the first separation attempt from the first iteration.

The ratio E^i is then treated as a comparison metric between the current state of separation in iteration i and between the starting point for the model. From this perspective, it was proposed that a link would be made between the ratio E^i and the cost C for the SVM from Equation 2.12. Intuitively this means that with a better separation level of the classes in the feature space, the penalty for making an error should be higher. Therefore, the cost for the current iteration (i) was calculated through:

$$C^{(i)} = E^{(i)} \times C^{(1)} \tag{3.25}$$

In this way, the SVM is punished more for a false sample as the separation of the classes improves.

3.5.2 Feature-Space Visualisation

In this chapter, a variation of the stochastic neighbour embedding (SNE) called "t-SNE" [81] is used to visualise the separation of classes in the latent feature space. t-SNE overpowers other visualisation techniques that have the tendency to cluster points at the centre of the visualisation map. t-SNE also has the ability to unmask a multi-scale structure in a single map. t-SNE converts the Euclidean distances between the different data points into joint probabilities and then uses gradient descent to minimise the sum of Kullback-Leibler divergences between all points. However, this cost function is not convex, which leads to different mappings with every run. This is not an issue for the visualisations in this chapter, because the focus is simply on the class spread in the latent feature space and not on the exact movement of points through the iterations.

3.6 Results

For consistency, the results shown in this chapter were all the outcome of tests carried out using the same splits of the datasets as described earlier in Section 3.4.1. Further, all experiments used the same image pre-processing techniques (Section 3.4.3) without any alteration. Also, during the training, and for every iteration, the within-class and between-class scatter matrices were calculated using Equations 2.15 and 2.16. The within-class $\mathbf{S}_W$

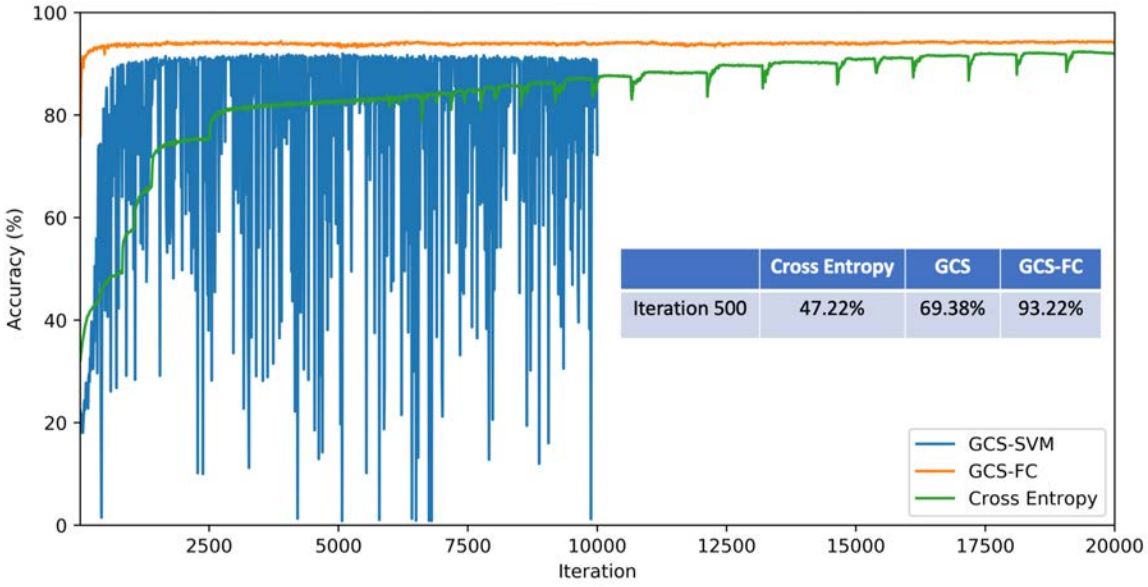


Fig. 3.2 Comparison between the accuracy on the CIFAR-10 test set between our proposed method using the ResNet-18, with SVM classification (GCS-SVM) and a two-layer fully connected network (GCS-FC), and the cross-entropy criterion.

and between-class $\mathbf{S}_B$ scatter matrices are then summarised using the norm for each of them $\|\mathbf{S}_W\|$, $\|\mathbf{S}_B\|$, those values are shown in Figure 3.5. The ratio defined in Equation 3.24 is then calculated and shown in Figure 3.6 for experiments with the cross-entropy criterion and others with our proposed method.

The trained models were tested continuously during the training process (every 10 iterations) to assess the model's prediction ability at various stages of the training process. With this approach, the test set was input into the model every 10 iterations and then the prediction of every sample was determined by the classification method specific to the approach. In such manner, the performance of the ResNet-18 and VGG-19 models is tested when trained using the following three approaches:

- The standard approach to classification training for neural networks, which uses cross-entropy and stochastic gradient descent.
- Our proposed method for gradual class separation with an SVM for classification (**GCS-SVM**). For this approach, it should be mentioned that test inputs were standardised before being passed into the SVM using a Standard Scaler (Equation 3.20), which is fitted using the training outputs of the last training iteration.

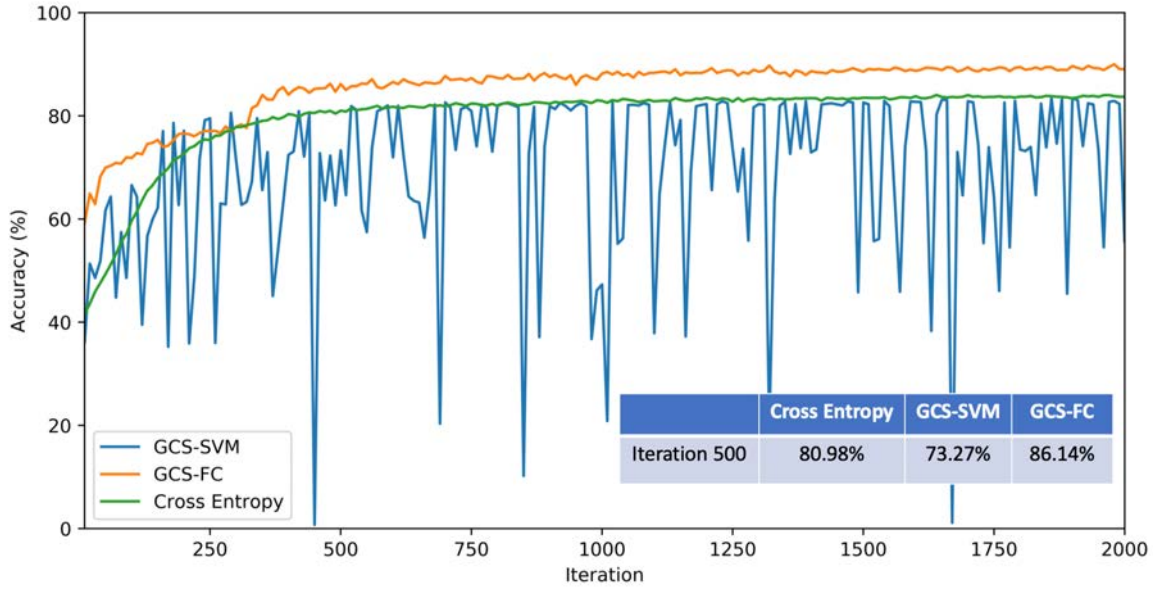


Fig. 3.3 Test accuracy comparison on the CIFAR-10 dataset between our proposed method (GCS) and the cross-entropy loss, where both were used to train the VGG-19 model.

- Our proposed method for gradual class separation but with a two-layer classification network that is trained independently using the cross-entropy criterion while taking the output features from the deep model as inputs (**GCS-FC**).

The above-mentioned approaches were tested on the CIFAR-10 dataset with both models, ResNet-18 (Figure 3.2) and VGG-19 (Figure 3.3). However, testing on the CIFAR-100 dataset was only carried out using the ResNet-18 architecture (Figure 3.4).

3.7 Discussion

3.7.1 Generalisation

Despite the ability of deep neural networks to memorise training sets, the aim behind any learning algorithm is to generalise. This aim was achieved in this scenario, where the model is able to extract features from unseen samples in a manner that is allowing the SVM ensemble to achieve state-of-the-art performance. This capability is also evident in the class distribution on unseen data produced by t-SNE and visualised in Figure 3.9.

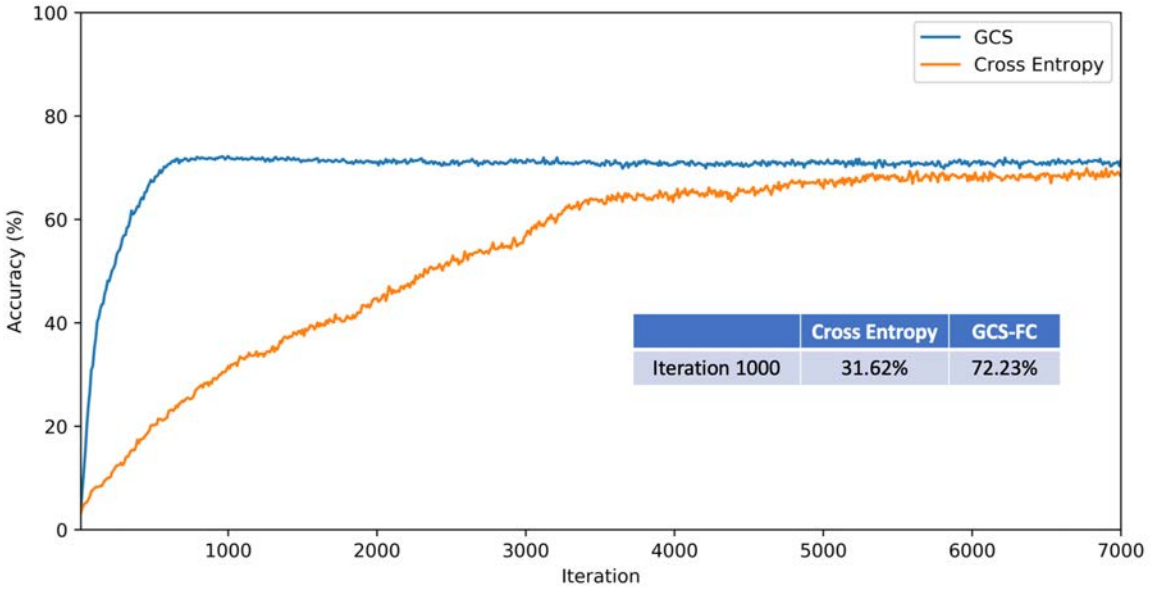


Fig. 3.4 Test accuracy comparison on the CIFAR-100 dataset between our proposed method, GCS, and the cross-entropy loss, both used to train the same ResNet-18 architecture.

3.7.2 Classification Accuracy

The effect of the GCS on the classification accuracy during the training of a Resnet-18 model on the CIFAR-10 dataset is clear from Figure 3.2. The same deep architecture was able to reach the benchmark classification accuracy in less than a quarter of the iterations it took with the cross-entropy criterion. Using the proposed GCS method, the model was able to exceed 90% accuracy before reaching the $1,000$ iteration mark, whereas it took the cross-entropy criterion more than $13,000$ iterations to exceed the same threshold. The Resnet-18 training process using our method was deliberately terminated at $10,000$ iterations, because there were no noticeable changes to the classification accuracy. However, as seen in Figure 3.2, the training process using the cross-entropy criterion was allowed to continue until $20,000$ iterations, where the model reached a performance on the same level as the model trained using our proposed method. It is worth mentioning that the performance for Resnet-18 trained using cross-entropy criterion agrees with the performance reported in the original paper by He et al. [36].

Interestingly, the slowdown in the improvement of the accuracy when training with GCS happens more suddenly compared to the cross-entropy criterion. This is exactly meant to be the attractive aspect of the cross-entropy criterion, which is to escape the slowdown in

training caused by fading gradients. However, it would be more attractive if we can reach the "best accuracy" much quicker, or even close to it and follow it with a fine-tuning using cross-entropy criterion.

The best accuracy reported from the Resnet-18 model trained using GCS was 92.09%, which is very close to that achieved by the model trained using the cross-entropy criterion, 92.23%. Also, in a similar manner to cross-entropy, GCS achieves a F_1 score that is very similar to its average accuracy, which is 92.08% compared to the 92.22% achieved by the cross-entropy. This indicated that GCS is not leaving any classes behind when increasing the spread of the classes in the latent feature space. This is also visible in Figure 3.7, where the confusion in prediction made by both models is similar. The errors are focused between three pairs of visually similar classes: (cat, dog), (ship and airplane) and (automobile and truck). This confusion is also reflected in the class distribution in the latent feature space, as captured by the t-SNE in Figures 3.8 and 3.9. As expected, the classes with similar visual features are neighbouring each other in feature spaces generated by both methods and intersecting in certain cases.

A question could be asked regarding the impact of replacing SGD with ADADELTA on the speed at which the model reaches its benchmarks accuracy. However, from [79], even though ADADELTA yielded lower error on the MNIST dataset than both SGD and ADAGRAD, the same pattern does not manifest.

The class separation patterns (Figures 3.5 and 3.6) reflect on the test accuracy patterns through the iterations (Figure 3.2). However, the exception from this pattern is that even though the separation ratio keeps rising for GCS-FC, the accuracy reaches a saturation level that it does not exceed. This is different from the pattern can be seen in the cross-entropy case, where the improvement in accuracy gradually slows down while the class separation ratio increases very slowly.

Accuracy Variance

The high variance in the classification accuracy of the predictions made by the GCS method can be attributed to the lax parameters that are passed to the SVM ensemble, which are used to accelerate the experiments.

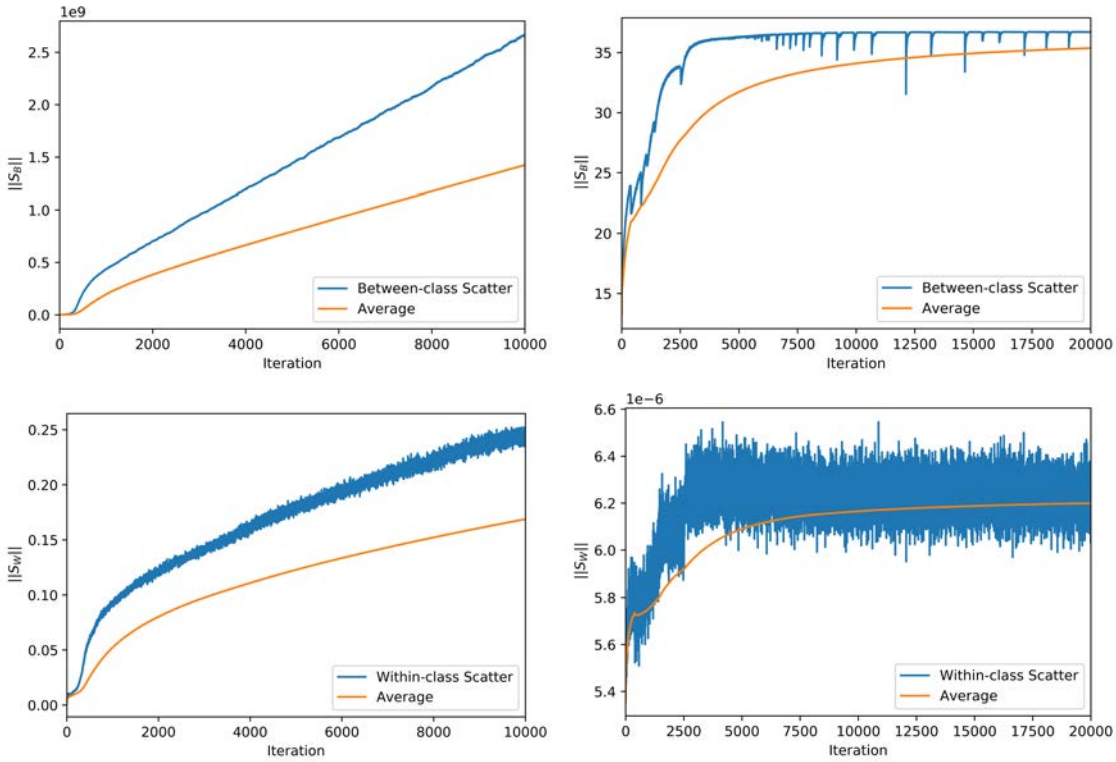


Fig. 3.5 Comparisons between the within-class scatter and between-class scatter of our method (left) and the cross-entropy criterion (right), while training the ResNet-18 on the CIFAR-10 dataset.

3.7.3 Computational Cost

Fitting the SVM ensembles adds no time to the training process, as it can occur in parallel. The feedback from the SVM fitting is not required for backpropagation, it is only the output features from the model that are needed to fit the SVM.

The computation overhead to compute the new ground truths in every batch is not substantial. The extra computation encompasses the calculation of the class centroids and then the distances for each centroid to be moved. The fact that a single ground truth is calculated for the entire class reduces the computation overhead significantly, while also targeting the reduction of the within-class scatter. Even though it might be an insignificant change in certain cases, but this method omits the need for the fully connected output layer along with the weights from the penultimate layer.

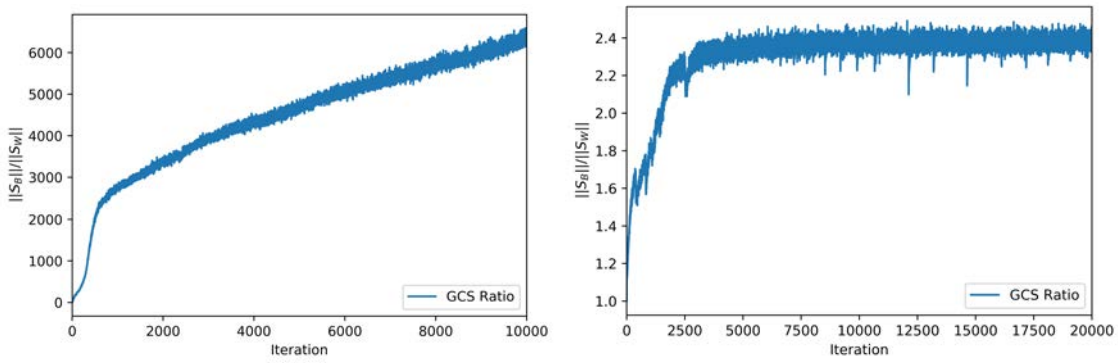
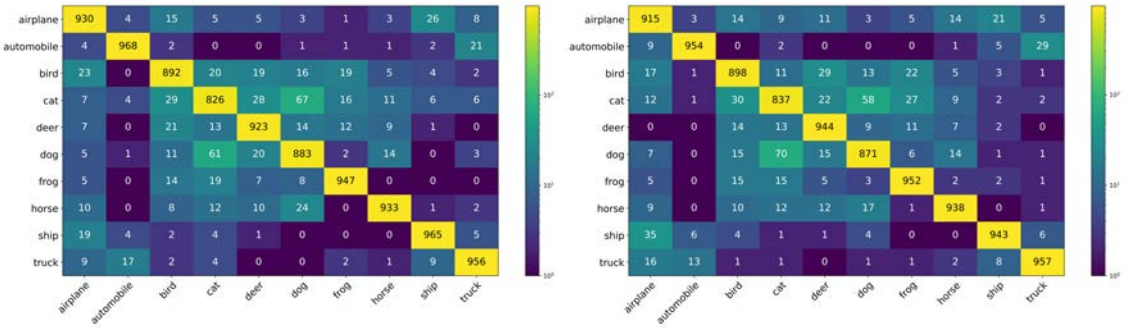


Fig. 3.6 A comparison of the separation ratio achieved during the ResNet-18 training iterations between our proposed method (left) and the cross-entropy criterion (right).



(a) cross-entropy (b) GCS

Fig. 3.7 Confusion matrices for the test predictions for the CIFAR-10 using Resnet-18.

3.8 Conclusion

The chapter has addressed an important research problem and one of the objectives of this thesis. It specifically focused on a new method to train neural networks to effectively learn representations towards classification. Our proposed method, GCS, was formally defined and tested using various deep models and benchmark datasets. Through this, our method was capable of achieving high validation accuracies that match the benchmark results in a substantially lower number of iterations. The results were compared using multiple models and datasets, and an in-depth analysis of the results was carried out.

From the work done in this chapter, many questions can arise for future works. Some of those questions can be concerning the potential behind progressive training of deep neural networks. While others can be surrounding the different separation criteria that can be used to train a deep model, by assessing the level of class separation at different levels of the model.

Additionally, the potential to introduce an unsupervised element to the training process to allow for training using unlabelled data.

The next chapter will explore the use of GCS to train deep neural networks for the classification of biomedical figures. The biomedical figure classification task presents some unique challenges to any classification training method.

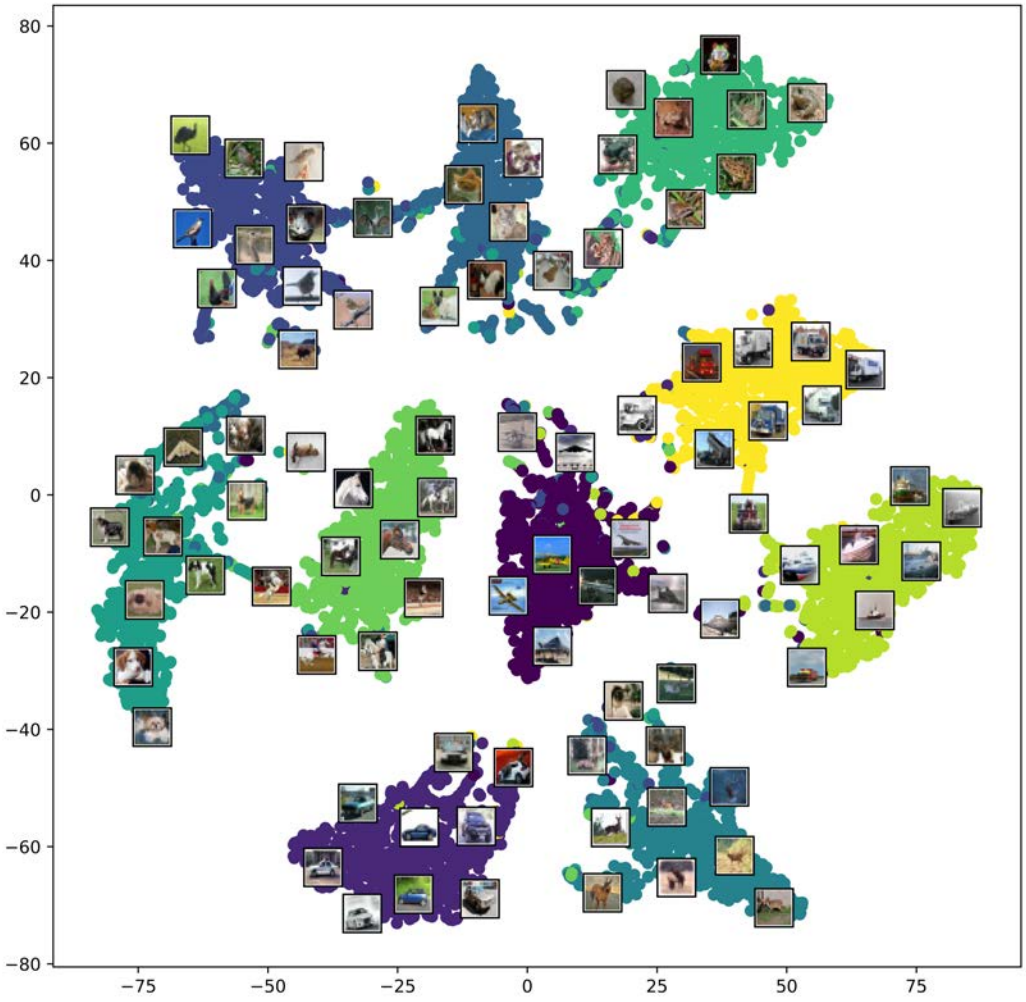


Fig. 3.8 Visualisation of the class separation in the reduced space using t-SNE on the unseen test set of CIFAR-10 after using Cross-Entropy.

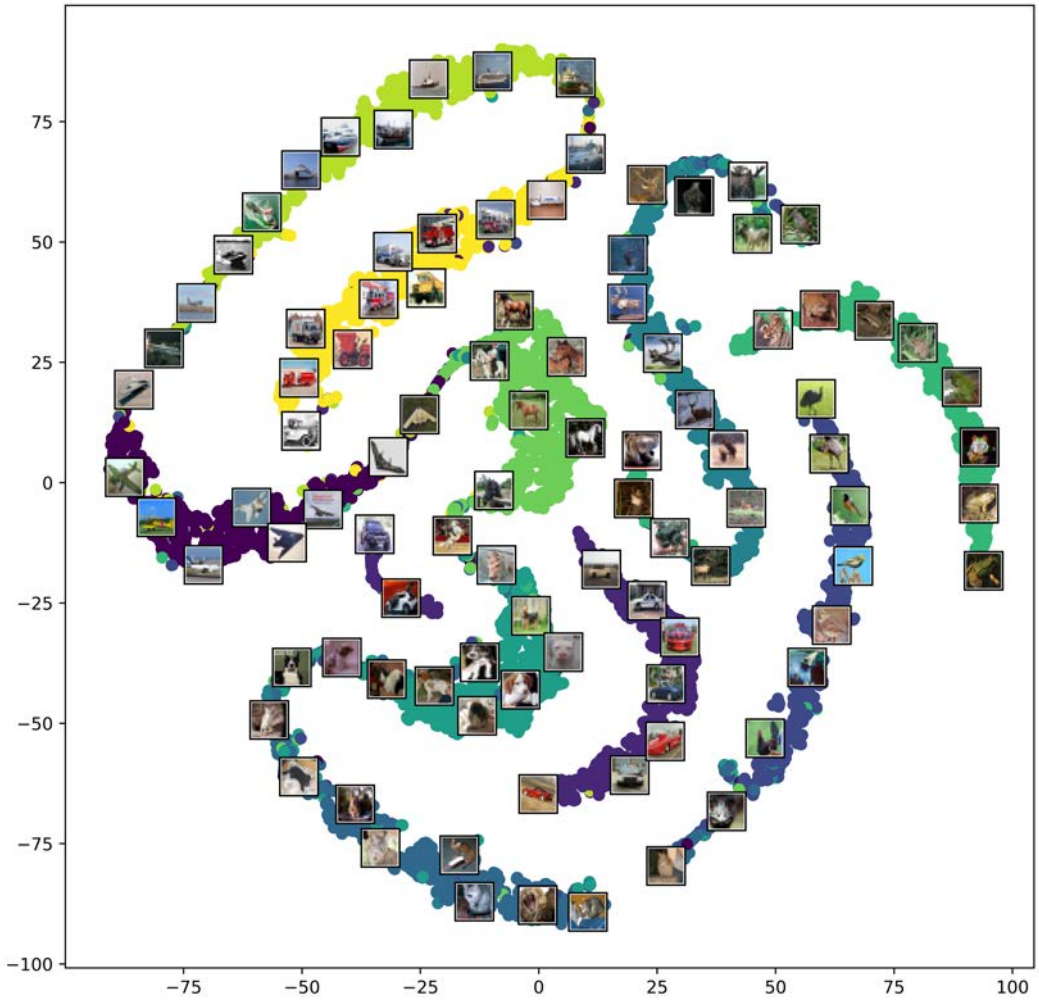


Fig. 3.9 Visualisation of the class separation in the reduced space using t-SNE on the unseen test set of CIFAR-10 after using GCS.

Chapter 4

Figure Classification

4.1 Introduction

Figure classification is one of the important steps towards mining information from biomedical figures [82]. As was detailed earlier in Section 2.2.2, biomedical figures were previously classified based on visual features, textual features, or a combination of the two. In this Chapter, the focus is the automatic extraction of features from the images (visual) of biomedical figures, moving away from feature engineering and into different aspects of deep representation learning.

This chapter discusses the biomedical figure classification task and addresses the different challenges that accompany it. Firstly, describing the dataset of biomedical figures that was used for the training and testing of the different models discussed in this chapter. The chapter then delves into the challenges specific to the classification of biomedical figures, including data class imbalance and the large number of classes. Addressing those challenges, the chapter describes the different models trained to classify biomedical figures using the ImageCLEF dataset, while addressing the arising challenges. Some of the work done in this chapter also builds on the gradual class separation method that was established in Chapter 3. Finally, the different results from the different proposed models are compared with each other and with the state of the art, while providing an in-depth analysis of the different results.

The following points summarise the contributions of this Chapter:

- Unsupervised training of a stacked deep autoencoder to extract visual features from biomedical figures towards classification;
- Introducing a multi-stage fine-tuning technique to enhance the classification performance, especially on classes that are less represented in the dataset;
- Developing a novel hierarchical ensemble of SVMs towards a better classification of biomedical figures;
- Supervised training of deep neural networks towards the classification of biomedical figures using cross-entropy criterion as well as the novel training method introduced in Chapter 3.

4.2 Dataset

The ImageCLEF dataset of biomedical subfigures was created as part of the medical figure classification task during the ImageCLEF 2016 challenge [82]. The dataset contains 10,942 biomedical figures extracted from open-source articles available on PubMed Central[1]. Each figure in the dataset has been manually classified by experts from the biomedical field into one of the 30 classes, which are listed in Section 4.2.1. As described in Chapter 2, this taxonomy was adapted from the work done by [28], which had 38 classes of which 31 classes appear in this dataset. The dataset was randomly split by the challenge organisers into 6,776 training images and 4,166 testing images. All the models described in this chapter were trained using the same data split prescribed by the challenge organisers.

Image Pre-processing

Biomedical figure images are of varying sizes and therefore to ease their input into our fully connected autoencoder model, all input images were resized to standard size of 100×100. This size was chosen to improve upon the training speed while also trying to keep as much of the important visual features as possible. The average size of the figure images in the ImageCLEF dataset is 303×231, as shown in Figure 4.1. However, it is also visible that the

[1] https://www.ncbi.nlm.nih.gov/pmc/

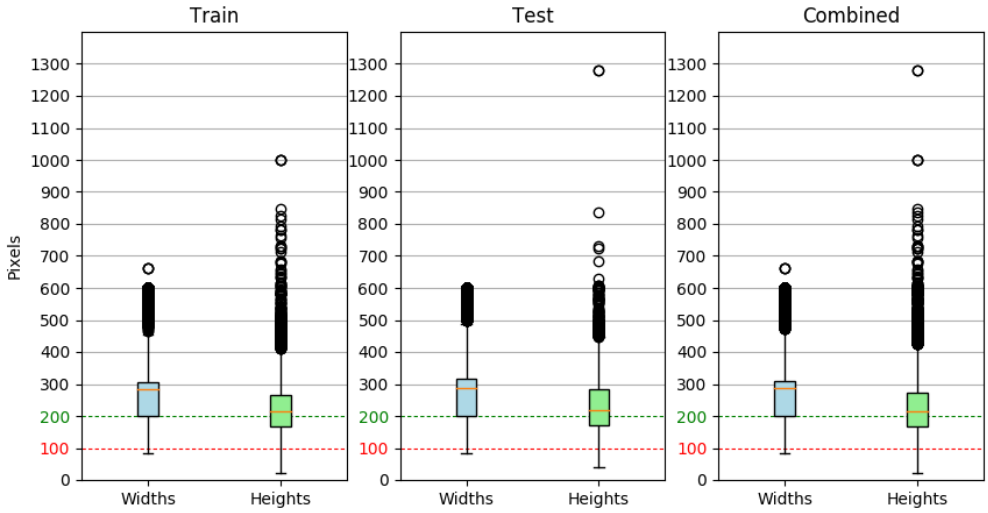


Fig. 4.1 The widths and heights of the training and testing images from the ImageCLEF dataset. The red line at 100 shows the size at which the images were resized for the SDAE, while the green one shows the sizes for the DCNN inputs.

average size for the testing set images is higher than the than that of the training set images, with 311×238 and 298×227, respectively. All the figure images within the ImageCLEF dataset are composed of three colour channels.

4.2.1 Taxonomy

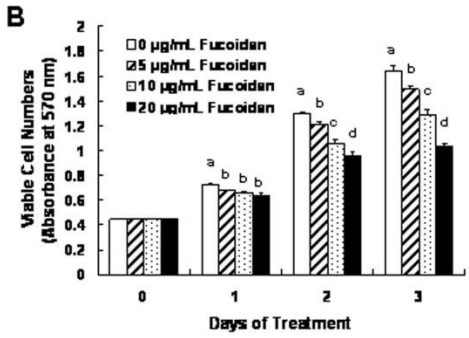
Each figure image within the ImageCLEF dataset can belong to one of 30 leaf classes, while also belonging to one of 6 parent classes. Each of the 30 leaf classes belong to one and only one of the 6 parent classes as shown in Table 4.1. For simplicity, each leaf and parent class have been allocated a specific number of letters, as shown next to each class in Table 4.1. Moreover, some of the parent classes, four of them to be exact, belong to another parent class, giving the taxonomy a further layer in the hierarchical structure. Figure 4.2 includes sample figure images for some of the classes from the taxonomy.

4.3 Challenges

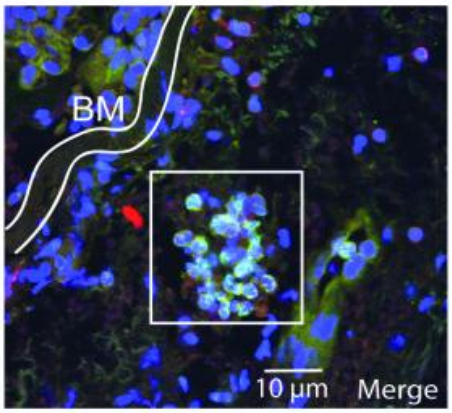
A few characteristics make biomedical figure classification a challenging task:

Table 4.1 A breakdown of the taxonomy used for biomedical figure classification [82].

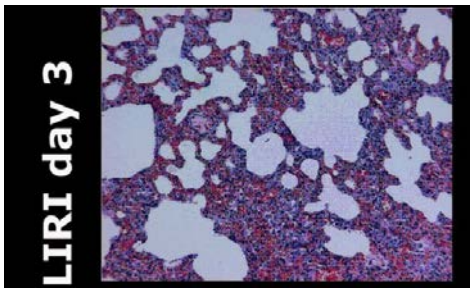
Biomedical Subfigures	
Diagnostic Images (D)	Generic Biomedical Illustrations (G)
- Visible light photography (DV): • Dermatology, skin (DVDM) • Endoscopy (DVEN) • Other organs (DVOR) **- Printed signals, waves (DS):** • Electroencephalography (DSEE) • Electrocardiography (DSEC) • Electromyography (DSEM) **- Microscopy (DM):** • Light microscopy (DMLI) • Electron microscopy (DMEL) • Transmission microscopy (DMTR) • Fluorescence microscopy (DMFL) **- Radiology (DR):** • Angiography (DRAN) • Combined modalities (DRCO) • Computerized tomography (DRCT) • Magnetic resonance (DRMR) • Positron emission tomography (DRPE) • Ultrasound (DRUS) • X-ray, 2D radiography (DRXR) **- 3D reconstructions (D3DR)**	• Tables and forms (GTAB) • Program listing (GPLI) • Statistical figures, graphs, charts (GFIG) • Screenshots (GSCR) • Flowcharts (GFLO) • System overviews (GSYS) • Gene sequence (GGEN) • Chromatography, Gel (GGEL) • Chemical structure (GCHE) • Mathematics, formula (GMAT) • Non-clinical photos (GNCP) • Hand-drawn sketches (GHDR)



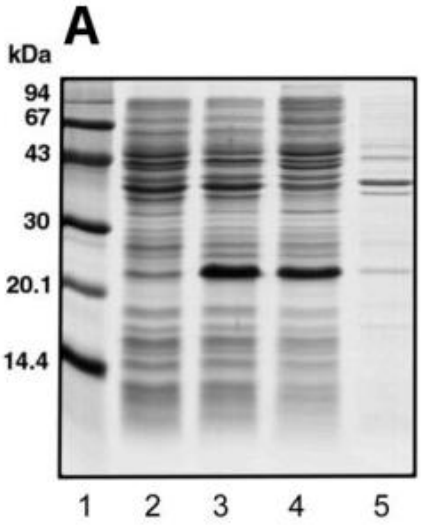
(a) Statistical figures, graphs, charts (GFIG)



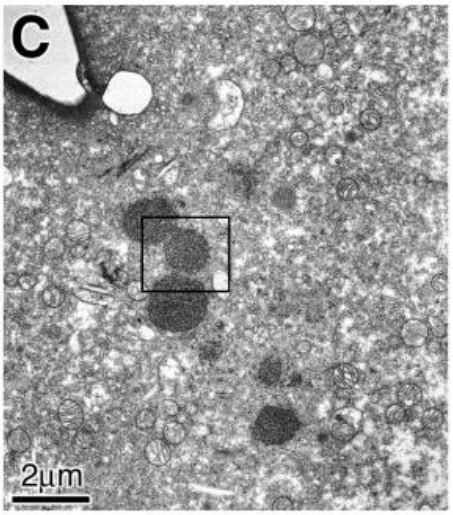
(b) Fluorescence microscopy (DMFL)



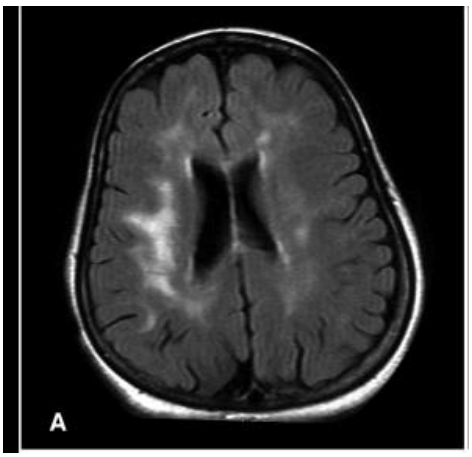
(c) Light microscopy (DMLI)



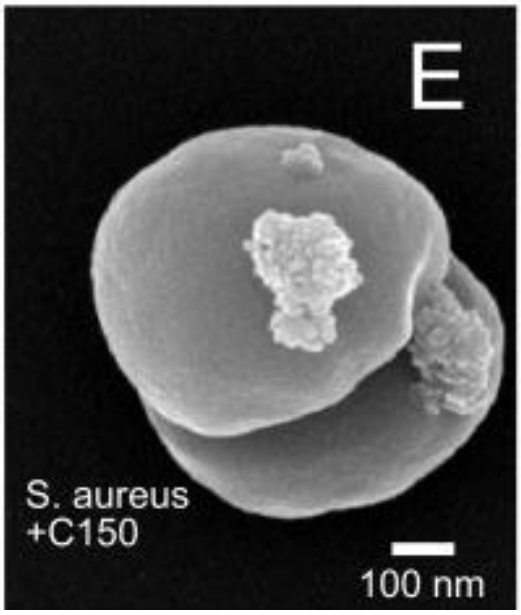
(d) Chromatography, Gel (GGEL)



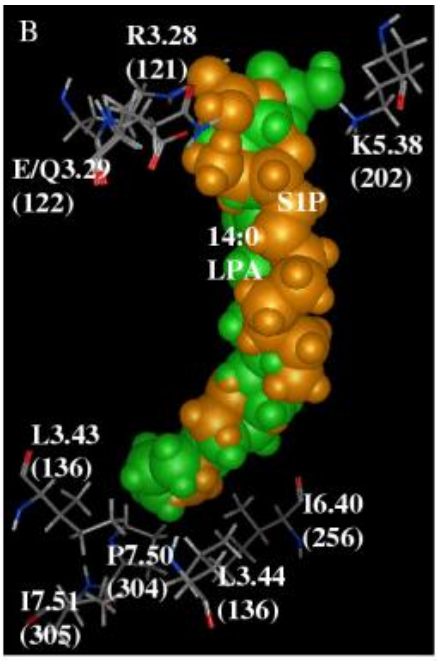
(e) Transmission microscopy (DMTR)



(f) Magnetic resonance (DRMR)



(g) Electron microscopy (DMEL)



(h) 3D reconstructions (D3DR)

(i) Gene sequence (GGEN)

Fig. 4.2 Sample training figures from the biggest classes in the taxonomy.

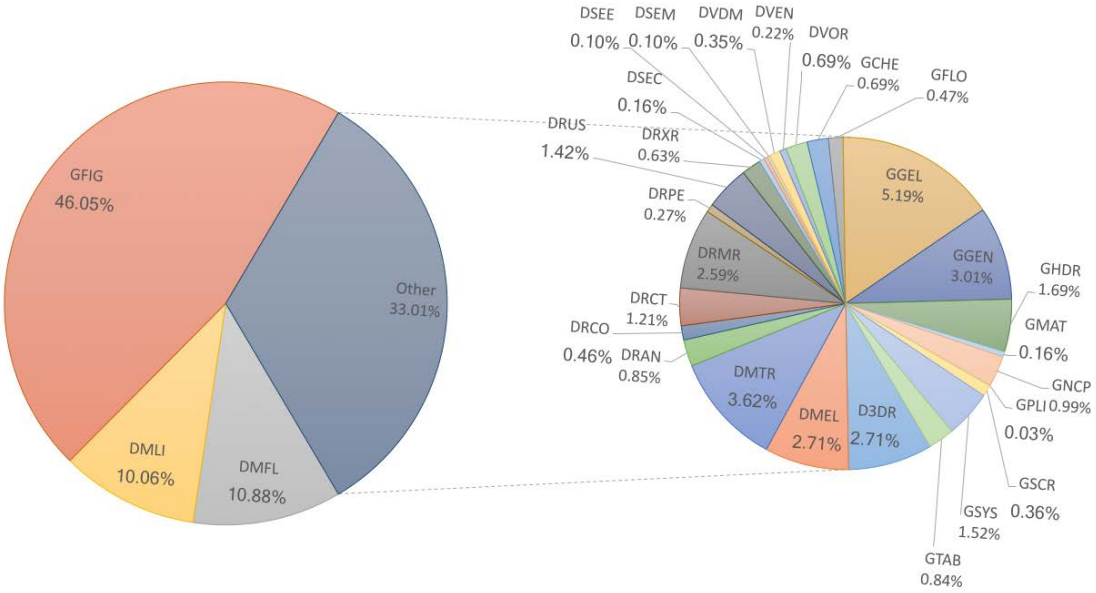


Fig. 4.3 The class distribution within the ImageCLEF 2016 dataset.

- **Huge class imbalance.** There is an enormous imbalance between the number of instances of the different classes (Figure 4.3). Specifically in the ImageCLEF dataset, the three largest classes (GFIG, DMFL, DMLI) amount for about 67% of the overall number of figures in the dataset, while the other 27 classes amount for the other 33%.
- **Small sample size.** The smallest 20 classes have less than a 100 training samples with one class having a single training sample (GPLI).
- **Large output space.** The large output space that contains 30 classes.
- **Subtle visual features.** The visual features that differentiate between some of the classes are very subtle, as demonstrated in Figure 4.4.
- **Low resolution.** The general poor resolution of figure images.

4.4 Feature Extraction

Prior to the rise of deep learning, many image classification problems were dependent on hand-crafted features. Such features were also used before for the classification of biomedical

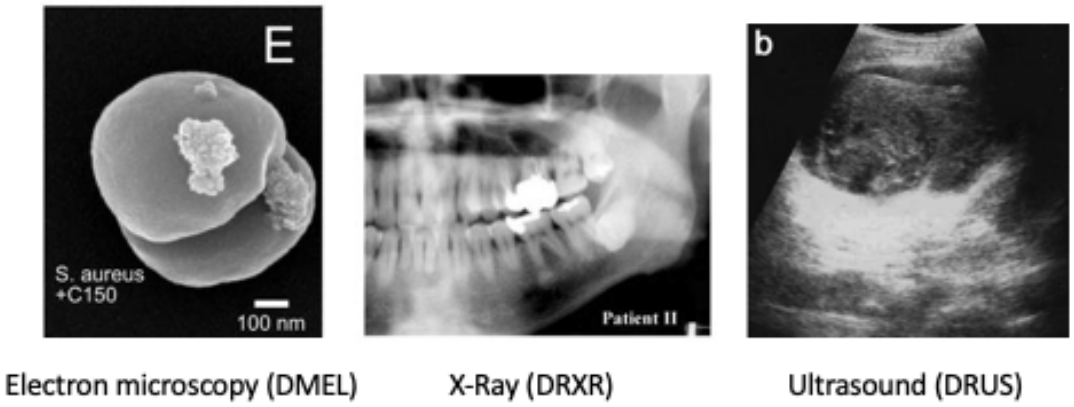


Fig. 4.4 The visual similarities between some of the different classes within the ImageCLEF dataset.

figures, which are described in detail in Section 2.2.2. This section describes the different methods we followed to automatically extract features from images of biomedical figures. Starting with a stacked deep autoencoder model and then deep convolutional neural networks. More details about the development of the Gradual Class Separation, GCS, method used to train the deep convolutional networks are described in Chapter 3.

4.4.1 Stacked Deep Autoencoder

As was previously discussed in Section 2.2.3, an autoencoder (AE) is a neural network that aims to learn a hidden representation $\mathbf{h}$ of an input distribution $\mathbf{x} \in \mathbb{R}^D$. Constraining $\mathbf{h}$ to a smaller space $\mathbb{R}^d$, where $d \ll D$, is one of the methods towards obtaining useful features in $\mathbf{h}$. The learning of this hidden representation $\mathbf{h}$ is achieved through the learning of two functions, an encoder $\mathbf{h} = f(\mathbf{x})$ and a decoder $\mathbf{r} = g(\mathbf{h})$, where $\mathbf{r}$ is the reconstruction. The encoder $f(\mathbf{x})$ is formally defined as:

$$\mathbf{h} = f(\mathbf{x}) = s_f(\mathbf{W}_e x + \mathbf{b}_e) \tag{4.1}$$

where s_f is the encoder's activation function, $\mathbf{W}_e$ is its weight matrix and $\mathbf{b}_e$ is its bias. The rectified linear unit (ReLU) is used as s_f:

$$ReLU(x) = max(0,x) \tag{4.2}$$

Due to their similarity to linear units, ReLU units allow for easier optimisation while also having large consistent gradients [34]. The decoder $g(\mathbf{h})$ on the other hand is defined as:

$$0\mathbf{r} = g(\mathbf{h}) = s_g(\mathbf{W}_d\mathbf{h} + \mathbf{b}_d) \tag{4.3}$$

where s_g is the decoder's activation function, $\mathbf{W}_d$ is its weight matrix and $\mathbf{b}_d$ is its bias. In this work, s_g was a sigmoid function that restricts the outputs to values between 0 and 1, which can be compared with the normalised grey-scaled input images. Training the AE is then a matter of finding $\mathbf{W}_e$, $\mathbf{W}_d$, $\mathbf{b}_e$ and $\mathbf{b}_d$ that minimise the following loss:

$$L(\mathbf{x}, g(f(\mathbf{x}))) \tag{4.4}$$

L being the loss function that measures the dissimilarity between $\mathbf{x}$ and $g(f(\mathbf{x}))$. s The challenge with constructing a deep autoencoder manifests in the fact that adding a layer to the encoder would mean adding another layer to the decoder, making the total number of layers grow at a magnitude of two. This growth in the number of parametrised layers makes training more challenging. Therefore, a greedy layer-wise approach [83, 84] was taken to train the stacked deep autoencoder (SDAE). The aim is to utilise the unsupervised initialisation to place parameters in a range of the parameter space that would lead to a good local optimum by local descent. Using this approach, both functions, $f(\mathbf{x})$ and $g(\mathbf{h})$, were learnt gradually by reducing the deep autoencoder into shallow autoencoders, which are simpler to train. Following the gradual training of the shallow autoencoders, the models were combined to form a stacked deep autoencoder (SDAE).

The SDAE model was trained in five stages in an unsupervised manner. Starting from an AE with 10,000 input and output neurons and a hidden layer of 7,500, where the 100×100 size images are flattened and fed into the model. In the second stage, the 7,500 output neurons for the first model were used as input and target for a new AE model with a hidden layer of 5,000 neurons. This was done in a recursive manner for the next three stages with

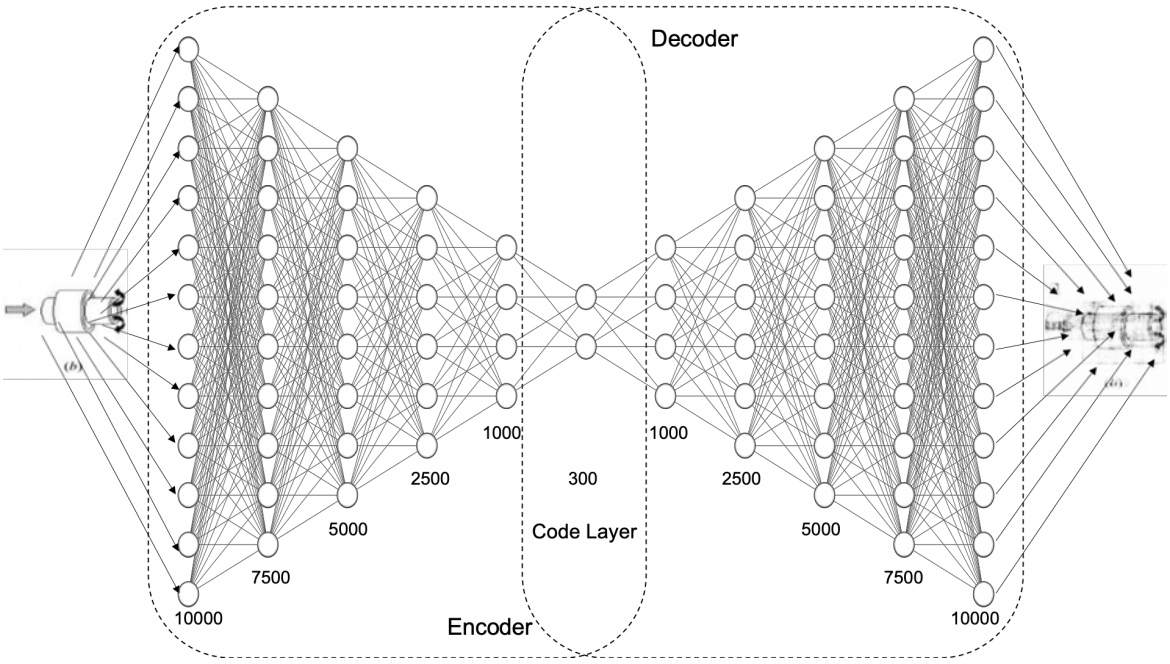


Fig. 4.5 The proposed SDAE model that was trained to reconstruct biomedical figures.

2500, 1000 and 300 hidden neurons, respectively. The encoders of those five AEs were then stacked to form the encoder of the SDAE and their decoders were stacked to form the SDAE decoder. The resulting SDAE architecture is shown in Figure 4.5. Each of the five AEs was trained by minimizing the b Binary Cross-Entropy (BCE) loss β between the model output o and the target t for N number of input figure images:

$$\beta = -\frac{1}{N}\sum t_i \log(o_i) + (1-t_i)\log(1-o_i) \tag{4.5}$$

Every activation function in the model was followed by a one-dimensional batch normalisation layer. Batch normalisation has proven to accelerate the training of neural networks by adjusting and scaling its layer activations [85].

Fine-tuning

Fine-tuning the SDAE model involved an initial unsupervised stage followed by a supervised one. The unsupervised fine-tuning was aimed at getting the different layers that were trained separately to work in cohesion. During which, the flattened figure images were set as inputs and targets for the model, while trying to reduce the same BCE loss β described in Equation

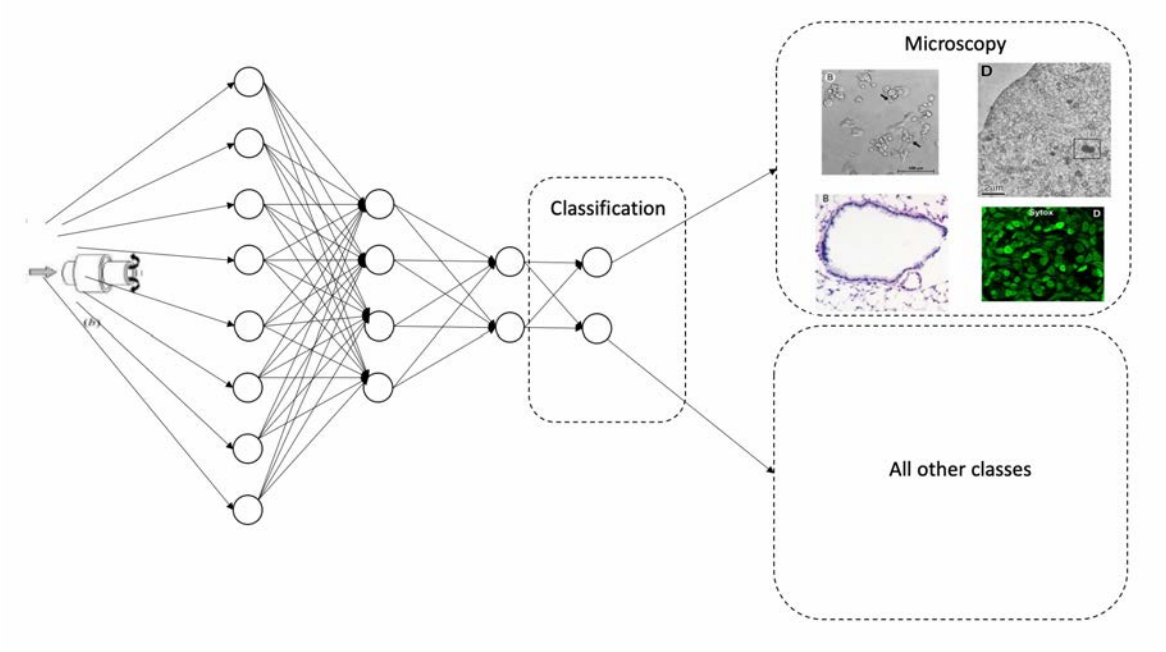


Fig. 4.6 The architecture of the model during the supervised fine-tuning stage.

4.5. The enormous class imbalance, described earlier in Section 4.3, was then tackled through detaching the decoder from the SDAE and replacing it with two fully-connected layers, with the second of the two having two output neurons. A different copy of the model was then trained to classify each class against the rest (Figure 4.6). The same one-vs-all training was done for all 30 classes from the ImageCLEF dataset. The classification fine-tuning was carried out by minimising the following cost function C, which combines the negative log likelihood criterion with a log Softmax function:

$$C = \sum_{i=1}^{N} w_c \Big(p_i + \log(\sum_{j=1}^{N} e^{p_j}) \Big) \tag{4.6}$$

where N is the total number of training samples, w_c is the weight for class c, c is the target class for input sample i and p is the probability of sample j belonging to class c. Weights were automatically assigned to each class depending on the class it is classified against to further enhance the model's ability to identify smaller classes. Finally, the last two layers were detached from the SDAE's encoder and the outputs from the 300 code neurons were used for classification as described later in Section 4.5.

4.5 Hierarchical Support Vector Machine

At a reasonable computational cost, SVMs can be fitted to compute the probabilities of each sample $\mathbf{x}_i$ belonging to any of the classes in the dataset. Thus, given k number of classes, the goal is to predict:

$$p_i = P(y = i|\mathbf{x}), \quad i = 1,\ldots,k \tag{4.7}$$

This is done following the estimation of the one-against-one pairwise class probabilities:

$$r_{ij} \approx P(y = i \,|\, y = i \; or \; j, \mathbf{x}) \tag{4.8}$$

Chang and Lin [45], in their LIBSVM implementation improved upon the work done by Lin et al. [86] to estimate this probability. More specifically, if $f(\mathbf{x})$ is the decision function that calculates the signed distance between $\mathbf{x}$ and the separation hyperplane, then it is assumed that:

$$r_{ij} \approx \frac{1}{1 + (e^{Af(\mathbf{x})+B})} \tag{4.9}$$

while A and B are estimated using the negative log likelihood (NLL) calculated from the labels and values of the training data. Also, as per the LIBSVM implementation a five-fold cross-validation is carried out to obtain the decision values before minimising the NLL. Cross-validation is used to avoid overfitting the model on the training values.

Following the calculation of r_{ij} for every pair of classes, a new objective is arrived at by Chan and Lin [45] using the second approach that is put forward by Wu et al. [87]:

$$\min_{\mathbf{p}} \quad \frac{1}{2}\sum_{i=1}^{k}\sum_{j:j\neq i}^{k}(r_j i p_i - r_i j p_j)^2 \tag{4.10}$$

where:

$$p_i \geq 0, \forall i, \quad \sum_{i=1}^{k} p_i = 1 \tag{4.11}$$

This was reformulated by Chang and Lin [45] to:

$$\min_{\mathbf{p}} \quad \frac{1}{2}\mathbf{p}^T Q \mathbf{p} \tag{4.12}$$

where:

$$Q_{ij} = \begin{cases} \sum_{s:s\neq i} r_{si}^2 & if\ i = j \\ -r_{ij}r_{ji} & if\ i \neq j \end{cases} \tag{4.13}$$

Chang and Lin [45] then uses the Lagrange multiplier β from the constraint $\sum_{i=1}^{k} p_i = 1$ to establish the following:

$$\begin{bmatrix} Q & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p} \\ \beta \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \tag{4.14}$$

Rather than going for a direct solution for Equation 4.14, Wu et al. devised an iterative method:

Algorithm 1: SVM probability estimates

Initialise $\mathbf{p}$ where $p_i \geq 0$, $\forall i$ and $\sum_{i=1}^{k} p_i = 1$
while *Equation 4.14 is not satisfied* **do**
  $t \leftarrow 1$
  while $t < k$ **do**
    $p_t \leftarrow p_t + \frac{1}{Q_{tt}}\Big(-(Q\mathbf{p})_t + \mathbf{p}^T Q\mathbf{p}\Big)$
    norm($\mathbf{p}$)
    $t \leftarrow t + 1$
  end
end
return p

The features extracted using the fine-tuned encoder from the SDAE model (Section 4.4.1) are used as training data to fit an ensemble of one-vs-all SVMs to output class probabilities [88]. Firstly, 30 SVMs were fitted, each tasked with one of the classification of one of the classes against the rest. Following that, and taking advantage of the hierarchical structure of the taxonomy (Table 4.1), a higher level ensemble of SVMs was fitted using encoders fine-tuned on the classification of those classes. This approach was aimed at dealing with the class imbalance, where the combined number of classes from the Diagnostic Images (D) parent class would potentially have a closer number of samples to the Generic Biomedical Illustrations class (G). The two-tier hierarchical structure of SVMs was then put to work together, where the class of an input image is determined by the maximum product of probabilities:

$$\max_{1 \leq n \leq m} \Big((p_1 \times p_2 \times \cdots \times p_l) \times p_n \Big) \tag{4.15}$$

m being the total number of leaf classes in the dataset, and l is the number of parent classes that class n belongs to.

4.6 Deep Convolutional Neural Networks

DCNN models have been widely used in computer vision for applications that range from object detection to medical diagnosis. The power of convolutional neural networks comes from their ability to retain spatial features, making them very suitable for such tasks. This is done through various stacks of filter kernels that are able to learn the most salient visual features. This is an important feature for the case of biomedical figure classification, where the various types of shapes may define a specific class of figures. For instance, it could be easy to identify charts, if the lines of the axes are identified.

Architecture

The ResNet architectures were chosen for this task as they have proven their ability on many classification problems in computer vision, which was also part of the tests in Chapter 3. The power of the residual networks originates from their skip layers, which were described with greater detail in Section 3.4.2. The feature extraction part of the ResNet model is made out of four main blocks. The number of blocks does not grow with the growth of the model, but the depth of each of the blocks increases.

More specifically, the ResNet-18 and ResNet-34 variations were chosen and their final layers were altered to accommodate the 2 class and 30 class problems respectively. It was deemed that the Resnet-18 is more than sufficient to carry out the binary classification between the Generic Biomedical Illustrations and the Diagnostic images parent classes. However, the ResNet-18 does not have the capacity to learn the features necessary to differentiate between all 30 classes of the ImageCLEF dataset. Therefore, the ResNet-34 mode was used for the classification of the 30 child classes.

Training

The first step to tackle the class imbalance was to calculate weights **w** for each class from the training set [89]. Thus during training, the aim was to minimise the following weighted cross-entropy (CE) loss for the output vector **o**:

$$loss(\mathbf{o}, c) = \mathbf{w}[c]\Big(-\mathbf{o}[c] + \log\big(\sum_j e^{\mathbf{o}[j]}\big)\Big) \tag{4.16}$$

where c is the target class. Stochastic gradient descent was used during training with a momentum set to 0.9 and a weight decay of 5×10^{-4}. The initial learning rate was set to 0.1 with milestones set at iterations 150, 250 and 350. At each milestone iteration i, the learning rate was changed to $lr_i = lr_{(i-1)} \times 0.1$.

The second attempt to strike a balance between the classes in each batch, a stratified sampler was used. The sampler combines random permutation generation with a stratified k-folds generator to ensure the variance between the batches during training.

Different data augmentation techniques were used to prevent overfitting on the training data and to provide an inflated number of samples of each class for the model, specifically aimed towards the smaller classes. The input training images were augmented using a combination of the following techniques: random horizontal flip, random rotation and random crop. When the random crop was not used, bilinear interpolation was used to resize the image down to the pre-defined input size. Resizing is the only function that was used for the input images of both the training and testing sets.

Gradual Class Separation

Our classification method proposed in Chapter 3, GCS, is put for another test in this chapter with a hugely unbalanced dataset. The GCS method is used to separate the features in the latent feature space, using the ResNet-18 and ResNet-34 architectures that were also trained using the CE criterion.

Minor alterations to the ResNet architectures were applied when training using GCS (Similar to changes done in Section 3.4.2); starting with changing the number of neurons in the output layers from the number of classes to the number of features to be extracted.

Consistent with the number of features in the penultimate layer of the ResNet architecture, 500 features were used while training the GCS method. This number was not changed from Chapter 3, where the ResNet-18 was used to classify the CIFAR-10 and CIFAR-100. The fact that 500 features were enough to distinguish between the 100 classes in the CIFAR-100 dataset indicates that the same number of features should be able to distinguish between the 30 classes of biomedical figures. Secondly, the Softmax functions are replaced with ReLU ones to allow for more freedom in the feature space. Finally, batch normalisation was applied at the output of the final layer to regularise the outputs. While training using GCS, the aim was to minimise the following loss function:

$$L(\mathbf{o}, c) = (\mathbf{o} - \mathbf{m}_c'')^2 \tag{4.17}$$

where $\mathbf{m}_c''$ is the target mean for class c that is calculated through Equation 3.12. The error is propagated back through the model at every batch, where the average of the batch losses. The regularisation parameter λ is initially set to 1.01 and then increased exponentially, setting the value of λ at iteration i to:

$$\lambda_i = \lambda^i - 1 \tag{4.18}$$

While training with GCS, ADADELTA [79] was used for optimisation with an initial learning rate of 0.999. The stratified sampler was also used in this case. However, unlike training for CIFAR-10 in Chapter 3, a single SVM was used instead of ensemble and the fitting of this SVM was done following the extraction of the features for the entire training set. This technique of fitting ensured that the SVM will get at least one sample from each class. The SVM used a second-degree polynomial kernel with a variable cost calculated through Equation 3.25. The kernel coefficient γ was set to 0.001, with a maximum number of iterations set to 10^6.

4.7 Evaluation Metrics

Accuracy is a simple metric that can be easily calculated to understand the performance of the classification model:

$$\text{Average } Accuracy = \frac{\sum_{i=1}^{C} \frac{TP_i+TN_i}{TP_i+TN_i+FP_i+TN_i}}{C} \tag{4.19}$$

where:

- TP_i is the total number of samples from class i correctly classified as class i.
- TN_i is the total number of samples from other classes correctly classified as not belonging to class i.
- FP_i is the total number of samples from other classes falsely classified as class i.
- FN_i is the total number of samples from class i falsely classified as not belonging to class i.

However, with highly imbalanced classes, the accuracy metric can become misleading because it does not accurately reflect the performance on the minority classes. On the other hand, the harmonic mean of precision and recall provides a better insight into the performance on such unbalanced classification problems, and is called F_1 score. Therefore, the results shown in this chapter are focused more on the F_1 score, but still show the overall classification accuracy.

$$\bar{F}_1 = 2 \times \frac{1}{\frac{1}{\bar{P}} + \frac{1}{\bar{R}}} \tag{4.20}$$

where:

$$\bar{P} = \frac{\sum_{i=1}^{C} \frac{TP_i}{TP_i+FP_i}}{C} \tag{4.21}$$

and:

$$\bar{R} = \frac{\sum_{i=1}^{C} \frac{TP_i}{TP_i+FN_i}}{C} \tag{4.22}$$

4.8 Results

All the models were tested using the same test set provided as part of the ImageCLEF dataset [82] without any alteration. In a similar manner to training, all images were resized to 200×200 before being input into any of the models, except for the case of the SDAE. Due to computational constraints at the time, it was infeasible to train the SDAE model with an input image larger than 100×100.

During training, all deep models were validated on unseen data every 10 iterations. ResNet-18 was tested on the two upper classes of the biomedical figure hierarchy, the Generic Biomedical Illustrations (G) and Diagnostic Images (D), for which results are shown in Table 4.2. Following that the testing was done on all 30 classes of the dataset, for which Table 4.3 summarises the results for the tested models, with a focus on the F_1 score (Equation 4.20). The results shown are the best performances captured from the model after being assessed using the same evaluation metrics described in Section 4.7.

- **E-SVM.** An ensemble of one-vs-all SVMs were fitted using the features extracted using the encoder from the SDAE. Features extracted through the same encoder were then used to extract features from the test samples and then classified using the SVMs ensemble.
- **FE-SVM.** An ensemble of SVMs was fitted and then tested using the features extracted from the fine-tuned encoder.
- **HFE-SVM.** A hierarchical ensemble of SVMs was fitted and then tested using the features extracted using the same fine-tuned encoder. The class of each test sample was determined using Equation 4.15.
- **ResNet34.** After passing the figure images through the model, the class of the images was determined by the neuron returning the highest probability from the Soft-max layer.
- **ResNet34-GCS.** After passing the figure images through the model trained using GCS, the class of the images was determined by a two-layer classification network that is trained independently using the cross-entropy criterion while taking the output features from the deep model as inputs.

Table 4.2 Summary of results for binary classification between Diagnostic Images (**D**) and Generic Biomedical Illustrations (**G**).

Class	# Samples		ResNet-18	ResNet-18-GCS
	Train	Test		
G	2775	1494	0.96	0.97
D	4001	2672	0.92	0.95
$\bar{F}_1$			0.94	0.96
Overall accuracy (%)			94.65	96.35

4.9 Discussion

4.9.1 Stacked Deep Autoencoder Features

It is clear from Table 4.3 that the ensemble of SVMs (E-SVM) was not able to classify any further than three largest classes (GFIG, DMLI and DMFL), even when using weights so the SVM would incur higher cost for misclassifying the smaller classes. The confusion matrix in Figure 4.7 reveals more about this, where the E-SVM is simply opting to classify all the test samples as GFIG, DMLI and DMFL classes, except for a single test sample that was falsely classified as DRMR. The imbalance in this dataset is also mixed with a large output space, which makes this a unique problem. However, the model was still able to achieve 62.65% overall average accuracy, but the clearer picture is given by the F_1 score, where the model was only able to achieve 0.07.

Introducing a classification fine-tuning stage (FE-SVM) on top of the unsupervised fine-tuning, managed to marginally improve the accuracy of some of the other classes. This did not just raise the overall accuracy to 63.62%, but also increased the F_1 score to 0.12. The one-vs-all classification fine-tuning stage for different copies of the encoder adds minimal computational overhead to the training process, because the combined unsupervised training contributes to the bulk of the learning. From Figure 4.8, it is apparent that the SVM ensemble started making predictions beyond the three largest classes (GFIG, DMLI and DMFL). Through this fine-tuning stage, each of the feature encoders was able to learn features capable of identifying a specific class against the rest, which fits in with the one-vs-all framework of the SVM. However, this fine-tuning stage was still not sufficient for the SVM ensemble to raise the performance on smaller classes, particularly GGEL and GGEN when considering that each of them has more than a 100 training samples.

Table 4.3 Detailed results (F_1 scores) on the different classes of the ImageCLEF 2016 test set, where the performance of the different proposed approaches is compared.

Class	# Samples		E-SVM	FE-SVM	HFE-SVM	ResNet34	ResNet34 -GCS
	Train	Test					
D3DR	201	96	0.00	0.15	0.11	0.53	0.63
DMFL	906	284	0.47	0.65	0.62	0.83	0.82
DMTR	300	96	0.00	0.06	0.06	0.49	0.54
DRCO	33	17	0.00	0.00	0.00	0.00	0.06
DRMR	139	144	0.00	0.44	0.39	0.73	0.75
DRUS	26	129	0.00	0.00	0.00	0.00	0.85
DSEC	10	8	0.00	0.00	0.00	0.00	0.00
DSEM	5	6	0.00	0.00	0.00	0.00	0.00
DVEN	16	8	0.00	0.00	0.00	0.00	0.23
GCHE	61	14	0.00	0.00	0.00	0.73	0.80
GFLO	20	31	0.00	0.00	0.00	0.00	0.23
GGEN	179	150	0.00	0.00	0.05	0.42	0.40
GMAT	15	3	0.00	0.00	0.00	0.00	0.00
GPLI	1	2	0.00	0.00	0.00	0.00	0.00
GSYS	91	75	0.00	0.00	0.00	0.33	0.37
DMEL	208	88	0.00	0.19	0.2	0.52	0.48
DMLI	696	405	0.57	0.52	0.55	0.88	0.82
DRAN	17	76	0.00	0.00	0.00	0.00	0.56
DRCT	61	71	0.00	0.26	0.29	0.00	0.73
DRPE	14	15	0.00	0.00	0.00	0.00	0.00
DRXR	51	18	0.00	0.00	0.00	0.22	0.19
DSEE	8	3	0.00	0.00	0.00	0.00	0.00
DVDM	29	9	0.00	0.00	0.00	0.00	0.17
DVOR	55	21	0.00	0.00	0.00	0.54	0.26
GFIG	2964	2085	0.86	0.87	0.87	0.94	0.94
GGEL	344	224	0.00	0.00	0.21	0.76	0.70
GHDR	136	49	0.00	0.09	0.13	0.14	0.32
GNCP	88	20	0.00	0.05	0.00	0.00	0.38
GSCR	33	6	0.00	0.00	0.00	0.00	0.35
GTAB	79	13	0.00	0.00	0.00	0.19	0.27
$\bar{F}_1$ score			**0.07**	**0.12**	**0.13**	**0.27**	**0.40**
Overall accuracy (%)			**62.65**	**63.61**	**64.30**	**78.27**	**78.99**

Taking advantage of the hierarchical structure of the biomedical figure taxonomy, the hierarchical ensemble of SVMs (HFE-SVM) was able to raise the overall accuracy further, while improving on the accuracy of the smaller classes. The effect of the hierarchical structure is evident in Table 4.3, specifically with GGEL and GGEN figures. In combination, the different fine-tuning stages and the hierarchical SVM ensemble were able to almost double the F_1 score while also improving on the overall accuracy. The HFE-SVM was able to improve on the accuracy of classes with over 100 training samples, while it struggled to improve upon the accuracy of smaller classes. The only class that did not follow the same pattern was the computerized tomography class (DRCT), which we hypothesize is due to the very similar visual features shared by the different images in both the training and testing sets (Figure 4.9).

Clearly, the training procedure for this hierarchical classifier is a complex one, which includes many fine-tuning stages. However, the initial unsupervised fine-tuning stage simplifies the one-vs-all hierarchical structure.

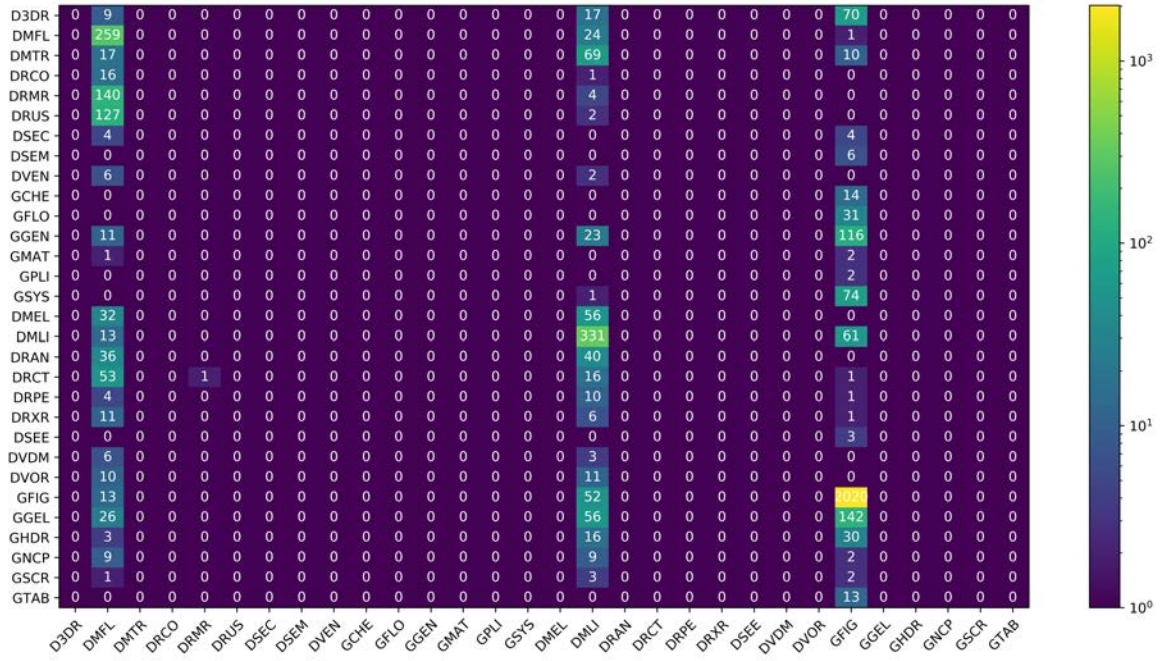


Fig. 4.7 The confusion matrix for the predictions made by the ensemble of SVMs (E-SVM) fitted using features extracted from the encoder prior to the fine-tuning stage.

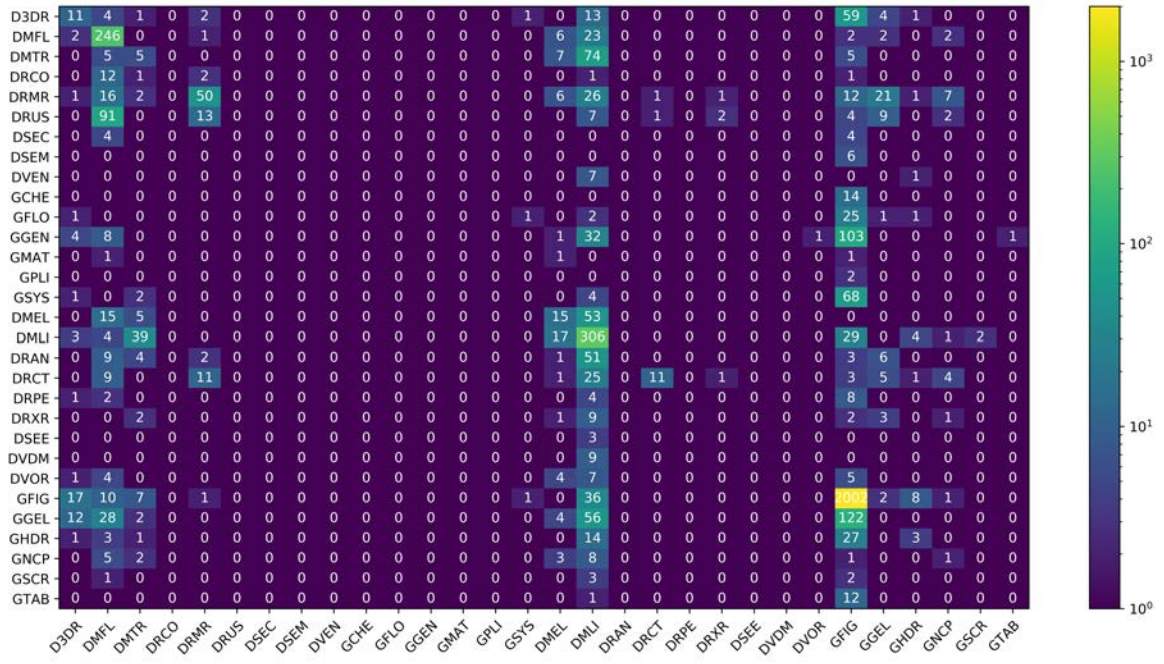


Fig. 4.8 The confusion matrix for the predictions made by the ensemble of SVMs (FE-SVM) fitted using features extracted from the fine-tuned encoder.

4.9.2 Deep Convolutional Neural Networks

A high accuracy along with a matching $\bar{F}_1$ score was achieved on the classification task between Diagnostic images (D) and Generic Biomedical Illustrations (G), as shown in Table 4.2. Those results also show the GCS training method proposed earlier in Chapter 3 surpassing the performance achieved by the same ResNet-18 architecture trained using CE with a considerable margin. Also, in a similar manner to the CIFAR-10 classification in Chapter 3, GCS was considerably faster to achieve higher accuracies. The model was able to reach an accuracy over 95% in under 200 iteration, whereas the CE criterion was only able to pass the 94% accuracy mark after 700 iterations.

Obviously, the ResNet-18 architecture used for binary classification (between Generic and Diagnostic figures) was not sufficient to perform the classification on the full taxonomy. Thus, ResNet-34 was chosen to perform the classification, while also comparing between models trained using CE and ones that are trained using GCS. This comparison is shown in the two rightmost columns of Table 4.3. A substantial difference is immediately noticeable between the results from the SDAE and the results from the ResNet-34 architecture. This is

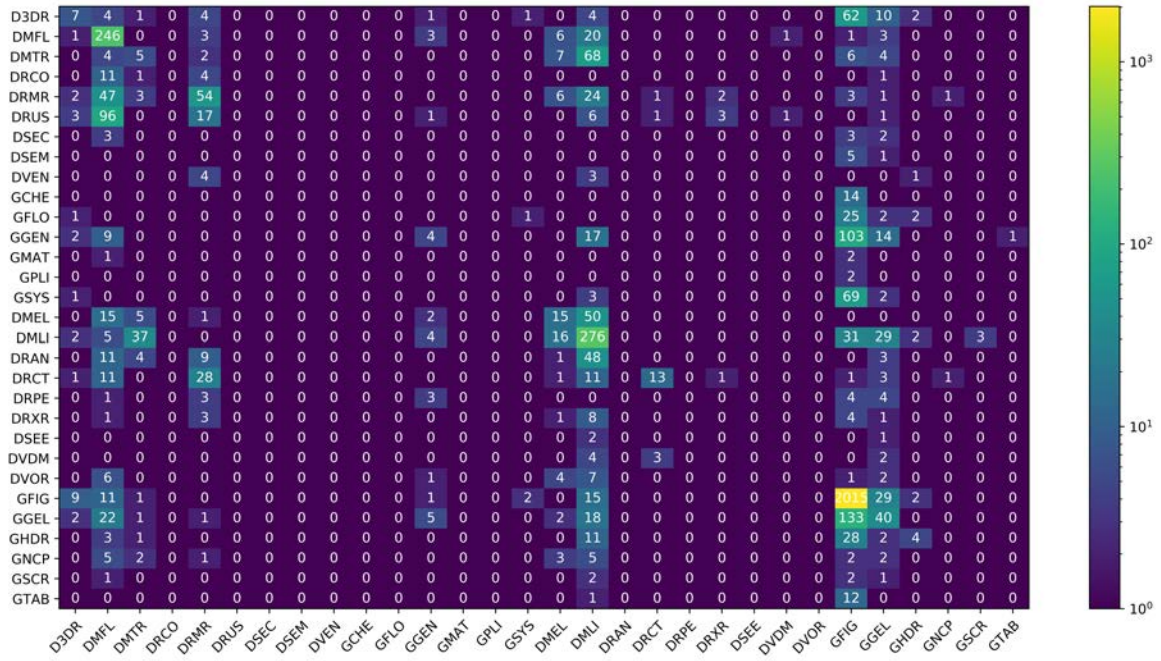


Fig. 4.9 The confusion matrix for the predictions made by the hierarchical ensemble of SVMs (HFE-SVM) using features extracted through the fine-tuned encoder.

due to the difference in the number of layers between the models, in addition to the effect of the spatial representations learnt by the convolutional filters. The convolutional layers also allowed to extend the input size of the figures from 100×100 to 200×200, as seen in Figure 4.1. Aside from the fact that 200×200 doubled the numbered of inputs, it is also closer to the average image dimensions and thus could retain more features from the original images.

Many classes previously unnoticed by the ensembles of SVMs trained using encoders' features were recognised with high accuracies, such as: GCHE, GSYS, DRXR and GTAB. Surprisingly, the hierarchical SVM was able to surpass the residual model, trained using CE, in identifying the computerised tomography class (DRCT). FE-SVM was also able to identify one of the test samples from the class for non-clinical photos (GNCP), where the different wrong attempts are also clear in the confusion matrix from Figure 4.8.

Following a similar pattern to previous experiments in Chapter 3, the model trained using GCS method was able to outperform the CE trained model, in terms of classification accuracy and efficiency. At the end of the training process for both models, trained using GCS and CE, GCS was able to achieve an overall accuracy of 78.99%, while the CE model performed slightly lower with 78.72%. ResNet-34-GCS was able to achieve this performance

much faster, where the ResNet-34-GCS was able to exceed 70% accuracy in 310 iterations compared to the 830 iterations it took for the CE ResNet-34. On individual classes, ResNet-34-GCS was able to surpass the ResNet-34-CE on most individual class metrics, except for: DMFL, GGEN, DMEL, DRXR, DVOR and GGEL. The ResNet-34-CE was able to marginally surpass the ResNet-34-GCS on the classification of those six classes with mid-range number of training samples. However, the standard ResNet-34 model was still unable to classify any samples of classes with lower than 50 training samples, which make up half of the total number of classes in the taxonomy (DRCO, DRUS, DSEC, DSEM, DVEN, GFLO, GMAT, GPLI, DRAN, DRCT, DRPE, DSEE, DVDM, GNCP and GSCR). ResNet-34-GCS still struggled with classes that have 15 or lower training samples, which is obviously due to the insufficient number of training examples to learn meaningful features to distinguish those classes (Figure 4.11).

The results achieved by the GCS are promising and show the potential to train deeper models for the biomedical figure classification task, similarly to what was done in [25, 26]. With GCS however, the training of such models can be done in a fraction of the time that it takes to train using SGD and CE criterion.

4.9.3 Class Imbalance

As anticipated, it is clear that class GFIG was a major cause of problems in the classification, especially with E-SVM, FE-SVM and HFE-SVM. However, given the depth of the ResNet-34 models, more representations could be learnt to differentiate between the large number of classes (Figures 4.10 and 4.11). It is also interesting to see that most of the confusion is happening between the GFIG class and other classes from the group of generic biomedical illustrations (G). This is mostly due to the more unique visual features differentiating the images of diagnostic images. As for the generic figures, as the name suggests, the images are of a wider variety and thus a classification model will go for the largest class aiming for a lower penalty.

The experiments on this dataset have proved that GCS is capable of dealing with class imbalance without the need to apply any weighting for the different classes. The GCS was also able of substantially improving upon the $\bar{F}_1$ score, which reached 0.4 when using GCS compared to 0.27 when using CE. This is consistent with the LDAs' resilience to

class imbalance [54], which was discussed earlier in Section 2.2.7. This further adds to the similarity between GCS and LDAs' underlying principles of maximising the between-class scatter while minimising the within-class scatter.

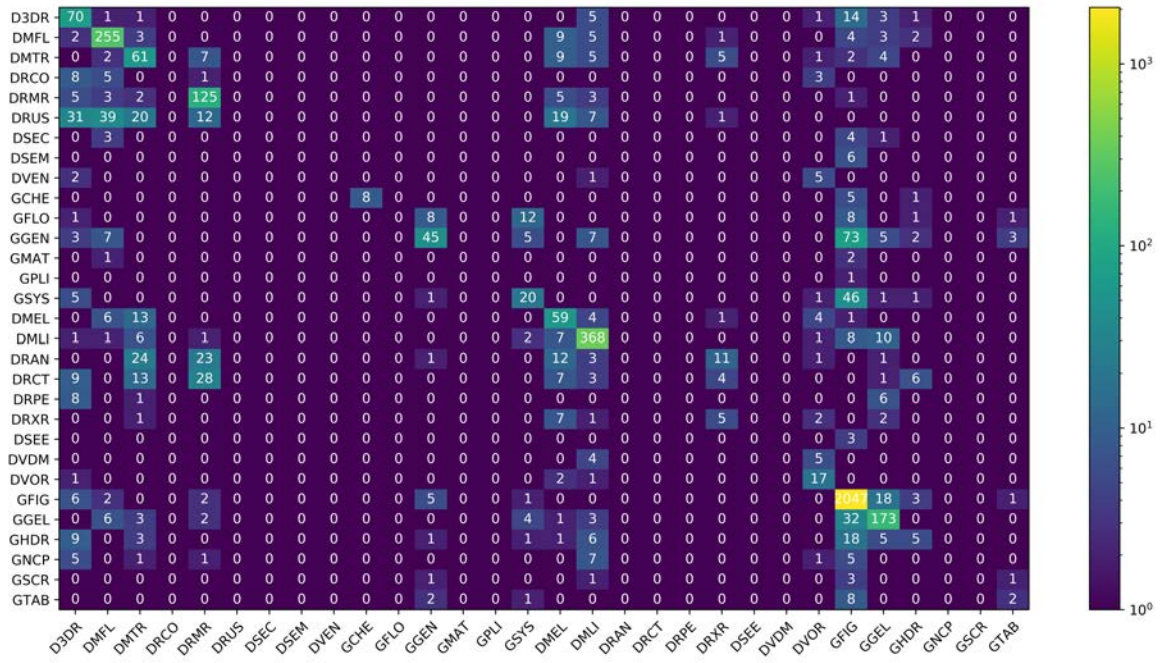


Fig. 4.10 The confusion matrix for the predictions made by the Softmax layer of the ResNet-34 model.

4.10 Conclusions

In this chapter, the task of biomedical figure classification was tackled through a variety of classification techniques. Novel classification techniques were proposed, some of which making use of the nature of the task at hand. Different approaches were also discussed, addressing the specific challenges with the biomedical figure classification task, such as class imbalance.

Firstly, a stacked deep autoencoder model was trained in an unsupervised greedy layer-wise manner to automatically extract features from biomedical figures. Different fine-tuning stages for the SDAE were proposed to tackle the imbalance between the classes in the dataset. Making use of the SDAE's encoders' features, a hierarchical ensemble of SVMs was proposed, benefiting of the hierarchy in the taxonomy of biomedical figures. The performance

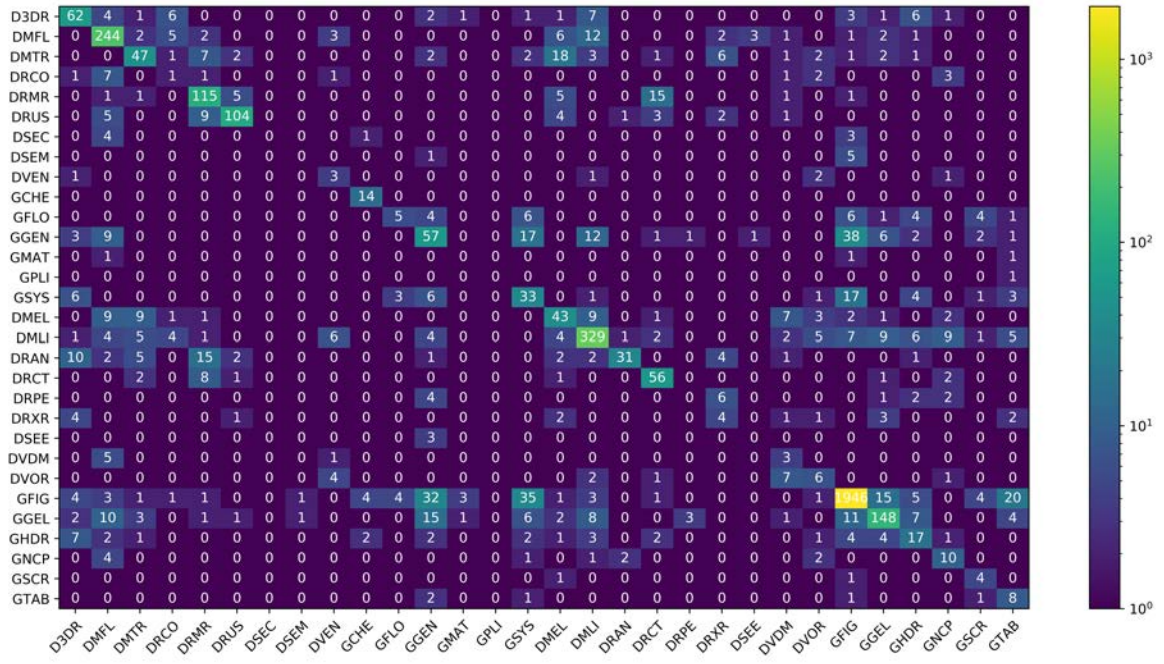


Fig. 4.11 The confusion matrix for the predictions made by the two layer neural network fitted using the features extracted through a ResNet-34 model that was trained using our GCS method proposed in Chapter 3.

of this model was compared to its standard counterpart as well as the fine-tuned models, but without the hierarchical SVM structure. The hierarchical model showed promising results with improvements on the classification performance, especially on certain classes that were entirely unnoticed by the other two models.

Deep convolutional models were then introduced, specifically a deep residual neural network (ResNet-18 and ResNet-34), which were first trained using the cross-entropy criterion. Unsurprisingly, the deep model was able to surpass the performance of the ensembles of SVMs fitted using a substantially shallower fully-connected neural network. Following that, the gradual class separation method, which was previously proposed in Chapter 3, was used to train a similar architecture to that of ResNet-34. The results achieved using the GCS trained ResNet-34 further support the results shown in Chapter 3, especially when dealing with data class imbalance.

The resultant classification of biomedical figures can be put towards an indexing engine, where the class of the figure would be part of the figure index. As mentioned in Chapter 1, this is the first step towards indexing using figures' visual features. The next step, in

Chapter 5, is for the localisation of text within biomedical figures towards extracting further information for figure indexing.

Chapter 5

Text Localisation

5.1 Introduction

This chapter delves into another stage from the process to extract information from biomedical figures, and that is to extract the text contained within the figure images. Extracting text from images is not a new task, however, the nature of biomedical images introduces new challenges that are discussed in this chapter. The chapter also proposes specific solutions that tackle those challenges, presenting a novel approach to localise the text within biomedical figure images.

Text localisation is an essential task towards the end-to-end extraction of text from images. Approaches to text localisation vary depending on the nature of the images containing the text. There exist two main categories of images from which text is extracted, scanned documents and natural scene images. However, biomedical figure images are different as they contain both types of images, where some categories of figures exhibit similar features to natural scene images and others exhibit features closer to scanned documents (Figure 5.1). More specifically, categories such as flow charts, tables and forms are closer to scanned documents with simpler backgrounds and higher contrast. While other categories, such as magnetic resonance, are closer to natural scene images with more complex backgrounds, smaller fonts and lower contrast.

Deep convolutional neural networks (DCNNs) have been widely used for computer vision applications, whether it is for classification or object detection. Many DCNN architectures

have been proposed to detect text from natural scene images and from scanned documents. Some DCNN models are dependent on proposal generation methods that elect specific regions from the input image for the DCNN to classify into two classes, one containing text and the other not. Proposal generation forms a performance bottleneck for text detection models, and therefore, the YOLO (You only look once) model [73] was proposed. The YOLO model is based on the idea of a single pass detection, thus saving on the performance needed for generating proposals and for the multiple classification tasks. In this chapter, a similar model is proposed that is also based on the single pass idea. However, to simplify the text localisation task and to make for finer detection that is required for the text detection task, the localisation task is simplified into a reconstruction one.

Dealing with the challenges posed by the nature of biomedical figures and the text regions they contain; specific training methods were utilised to overcome such challenges. A pre-training stage for the model was the first thing to be introduced using a large dataset of natural scene images with synthetic text. The second measure to deal with the challenges was introducing specific data augmentation techniques that target the different challenges ranging from random rotation to colour inversion.

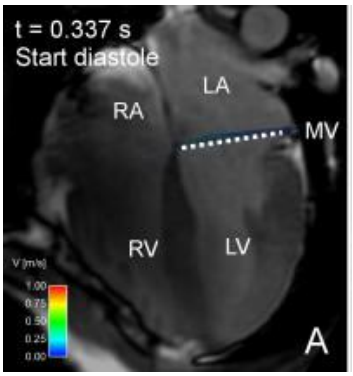
The results from the proposed model are thoroughly analysed to determine the model's ability to deal with the different text detection challenges in biomedical figures. The analysis is then used to draw a discussion, aided with the reconstruction samples, while future works are identified.

5.2 Datasets

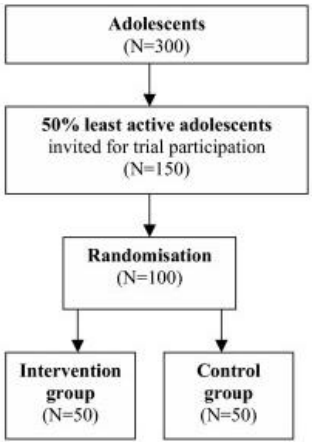
5.2.1 DeTEXT Dataset

Released in 2015, the DeTEXT dataset [58] of biomedical figures was the first, and still is, the only publicly available figure-text dataset. It was adopted by the ICDAR2017 robust reading challenge on the end-to-end extraction of text from biomedical figures [90]. Collected from 288 open-access scholarly articles randomly selected from PubMed Central[1], DeTEXT contains 500 biomedical figure images encompassing 9,308 text regions. Following the data

[1]https://www.ncbi.nlm.nih.gov/pmc/



(a) Magnetic resonance biomedical figure with complex background.



(b) A flow chart figure with a simple white background.

Fig. 5.1 Comparison between types of backgrounds for biomedical figures.

split proposed by [58], 100 figures were allocated for training, 100 for validation and the last 300 for testing. Each figure in the dataset is accompanied by a ground truth file that contains the different text regions in the image, with the following details for each text region:

- The anticipated difficulty of extracting the text from the region, which is based on the image quality and the type of text. Each region can be labelled as one or a combination of the following: normal, small, blurry, colour, short, complex background, complex symbol and specific text.
- The bounding box for the text region defined by the coordinates of its four corners.
- The text-string contained within the text region.
- A Boolean that is true when the region's bounding box is oriented (i.e. not horizontal) and false otherwise.

5.2.2 SynthText in the Wild Dataset

SynthText in the Wild [91] contains 800,000 synthetically generated images, each having a number of differently styled words embedded into a set of background images in a manner that accounts for the 3-dimensional scene geometry. Due to the substantial size of this dataset, it has been used for pre-training deep models towards end-to-end text extraction. Every

image in the dataset has an accompanying ground truth file that contains the text instances in the image, each with its text-string and bounding boxes at both word and character levels.

5.3 Challenges

Using the statistics gathered by Yin et al [58] for the different categories from the DETEXT dataset, the challenge is clear with only 37.8% of the collected text regions classified as "normal". The remainder 62.2% of the "abnormal" text regions were classified as one or a mix of the following challenges:

1. **Small text size.** Due to the limited space available for figure-text, authors often tend to make the font smaller if they need to insert text into a figure. 26% of the text regions from the DETEXT dataset were classified as small text regions.

2. **Short text.** Oftentimes, short text is used in figures, which is usually a character or two. It is mostly used to either split the figure into subfigures that can be easily mentioned in the body of text, or in the axis of charts to represent numerical values.

3. **Complex symbols.** Chemical and molecular formulae and abbreviations are not rare in biomedical figures. Complex symbols are occasionally contained within short text regions.

4. **Specific text.** This can be either a gene sequence or a linked term. End-to-end extraction of gene sequences is especially difficult due to the variable spacing between the different characters and the different colours used in certain cases.

5. **Oriented text.** Any text region that is not horizontal is considered to be oriented. Over 9% of the text regions in the DETEXT dataset are oriented, which are contained within 53.6% of the collected figures. The fact that oriented text rarely occurs in a figure without horizontal text regions increases the challenge.

6. **Coloured text.** Coloured text is usually used in figures to try to highlight the text and make it visible, especially on complex backgrounds. The association between coloured text and complex backgrounds can be clearly noticed in Figure 5.2.

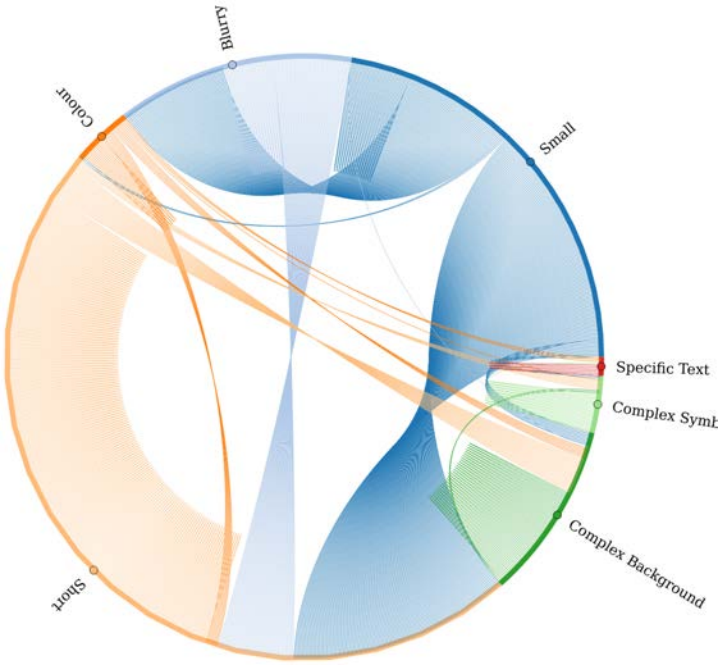


Fig. 5.2 The associations between the different categories of text regions within the DETEXT dataset.

7. **Complex background.** In biomedical images, text is sometimes intertwined with images when experimental results or objects are described.

8. **Blurry image.** Caused by file size limitations, compression and figure mishandling, 12% of the text regions in the DETEXT dataset are blurry.

A single text region can have one or more of the above-mentioned challenges at the same time. The relationship between the occurrence of the different challenges for the same text regions is laid out in Figure 5.2.

5.4 Proposed Method

5.4.1 Architecture

Autoencoder models [31] have been an attractive choice for unsupervised pre-training for classification. However, the text localisation task does not require any dimensionality reduction, therefore, a model was developed (see Figure 5.3) that maintains the size of the inputs

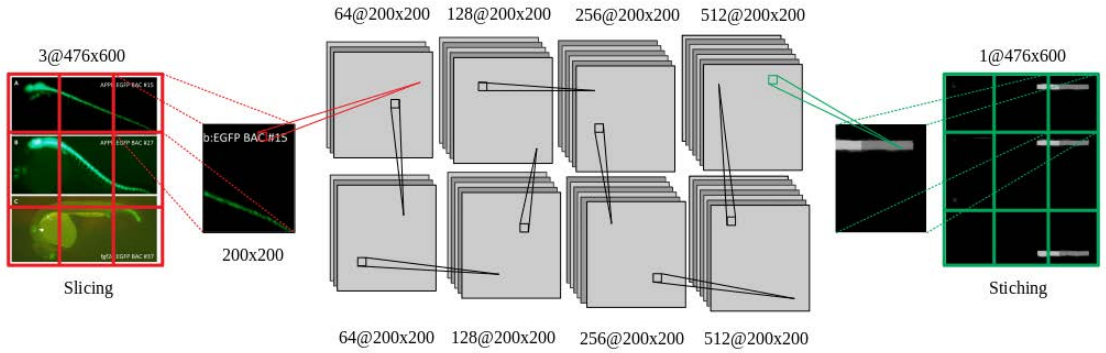


Fig. 5.3 The proposed model architecture for text localisation.

throughout. Such architecture decreases the chance of losing features during dimensionality reduction. In turn, this simplifies the text detection problem into a reconstruction one, where each input image is reconstructed into a single channel image. The reconstructed image would simply be an all-black image except for white areas covering predicted regions of text.

The proposed model maintains the input image size through the layers by following a few strategies. Firstly, padding was used for each convolutional layer starting with a padding of 5, and then reducing along with the filter size towards the output layer, where it finally becomes 1. Secondly, pooling layers were omitted from our architecture, because as stated, our goal is to maintain the dimensionality of the data through the model. The model employs eight convolutional blocks, each containing a convolutional layer proceeded by a Rectified Linear Unit (ReLU, Equation 5.1) and a 2-dimensional batch normalisation layer [85]. The model's eight convolutional layers can be broken down into four "twin" layers, as every two consecutive layers have exactly the same parameters. Instead of ReLU and batch normalisation, the model's output layer uses a sigmoid activation function to output values between 0 and 1, which is the probability of each pixel belonging to a text region [92].

$$ReLU(x) = max(0,x) \tag{5.1}$$

5.4.2 Data Augmentation

Data augmentation was used to deal with the comparatively small number of biomedical figure images available in the DETEXT dataset. An array A, containing sets of transformation functions T, was used to augment input images during the pre-training and fine-tuning stages. For each image in any given batch during both training stages, a set of transformation functions T was picked at random from A. Once the set T was selected, the transformation functions contained within T were applied to the input image in the exact order. The transformation functions used were: random scaling, colour inverse, random rotation, random crop and random sized crop. Following the transformation of input images, some transformations required the changes to also be applied to the target, random scaling and random cropping are examples of this.

The transformation functions were especially selected to address specific challenges posed by text localisation from biomedical figure images (Section 5.3):

- **Random scaling and random sized crop.** Random sized crops and random scaling of input images were targeted at dealing with the diverse range of text sizes that exists in biomedical figures.
- **Random rotation.** By randomly rotating training images, it is possible to deal better with the localisation of oriented text regions.
- **Colour inverse.** This was aimed at dealing better with the variation of background colours and more complex backgrounds.

5.4.3 Training

The proposed model was first trained using the SynthText dataset (Section 5.2.2), which contains a very large number of training samples. Considering the nature of the images in this dataset, which is closer to natural scene images, the model could learn features that would ease the learning on biomedical figure images. The model was pre-trained for 3 iterations through minimising the following pixel level binary cross-entropy loss (C):

$$C(o,t) = -\frac{1}{N}\sum t_i \log(o_i) + (1-t_i)\log(1-o_i) \tag{5.2}$$

where N is the total number of pixels, o_i is the model's prediction for pixel i and t_i is the target for that pixel i. Adagrad [93] algorithm was used for optimisation with the initial learning rate set to 0.1 and learning rate decay of 10^{-4}. Batch training was used with a batch size of 20.

During training, ground truth images were generated on the fly, in the manner shown in Figure 5.4b, using the ground truths provided in the datasets (Figure 5.4a). All training images were resized to 200×200 using to fit into the input convolutional layer. Bicubic interpolation was used for scaling to preserve as much smoothness in the resized image. While resizing the input images, the aspect ratio was preserved from the original image through using the following combination of resizing and scaling:

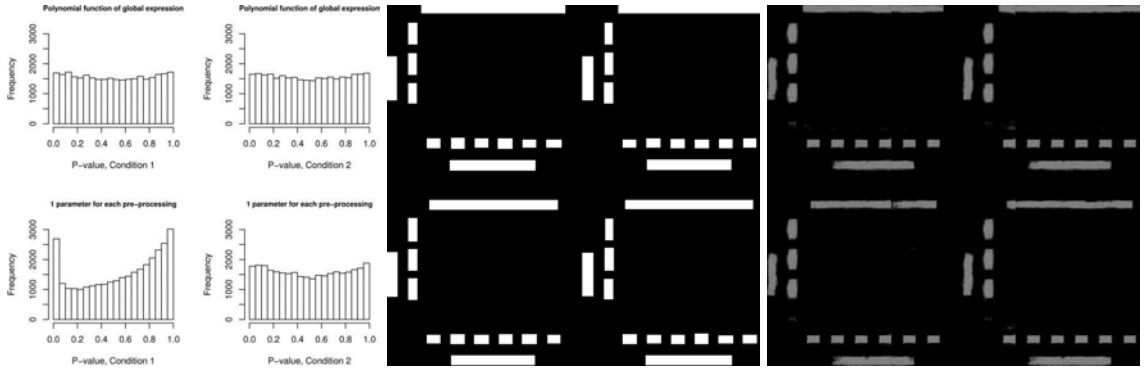
- If both dimensions are larger than 200, each is resized according to the relevant ratio.
- If one of the dimensions is larger than 200, the larger dimension is resized to 200, while the smaller dimension is padded to reach 200.
- If both dimensions are smaller than 200, both dimensions are padded to 200.

Following pre-training, the model was fine-tuned using the DETEXT training set for 5×10^4 iterations. In a similar manner to the pre-training stage, the fine-tuning was aimed at reducing C from Equation 5.2. Adagrad [93] was also used for this stage, however, and considering it is a fine-tuning stage, the learning rate was initialised to 10^{-3}, with a lower learning rate decay of 10^{-7}. The same on-the-fly data augmentation techniques were used during both stages of the training process, which are described in detail in Section 5.3.

5.4.4 Image Processing

During testing, input images are treated differently, where they are not simply resized or padded in the same way used for training. Two main strategies are followed for test images:

- The first addresses images smaller than a threshold that was set to 600×600, which were sliced into 200×200 sized images. This meant that each input images larger



(a) Input image (b) Generated target image (c) Model output

Fig. 5.4 Training sample from DeTEXT dataset.

than the specified threshold would be sliced into $n = \lceil \frac{W}{w} \rceil \times \lceil \frac{H}{h} \rceil$ slices, with W and H being the width and height of the input image respectively, while w and h are the predefined slice dimensions that were set to 200×200. To utilise the overlap area for better predictions, the remainder area is distributed over the n slices to allow for a vote between the slice predictions.

- The second strategy was set to deal with images that are larger than the 600×600 threshold. In this case, and in a similar manner to the pre-processing done on the training images, a combination of bicubic interpolation and padding were employed to get the image dimensions below the threshold. Following this reduction, the first strategy is used to slice the image into n slices and input it into the model.

The resulting slices of the test image are then batched together and passed through the model in a single go. The specified maximum size of 600×600 was chosen to ensure that the test batch size cannot exceed 9. Once the test batch is put through the model, the different output slices are stitched back using a simple mapping method that keeps track of the location of each slice. As for the overlapping areas between the different slices, the values are averaged out to gain a more informed decision.

As the output layer of the proposed model uses a sigmoid layer with outputs between 0 and 1, the output reconstructed image is a single channel image (Figure 5.4c). Therefore, the regions containing values closer to 1 are text regions, while other regions containing lower values are text-free. Following that, experiments were carried out to identify a good threshold point that can distinguish between text regions and noise. This led us to set a threshold value

of 0.3, under which the values are considered noise and above which it is considered to be a text region. Using a simple thresholding function, the reconstructed image is transformed into a binary image. The next step was to find the contours that separate the contrasting black and white regions using the method proposed by Suzuki and Abe [94] for border following. Finally, a simple noise filtering technique was employed by setting a minimum size for the detected text regions to 3×3. Even though biomedical figures contain a substantial number of text regions that contain only "short" text, which is normally a character or two, those are still normally of a size larger than 3×3.

5.5 Evaluation Metrics

Assessing the text detection task is similar to the assessment of a binary classification problem, one class being the actual text regions and the other being non-text regions. However, there have been different metrics that differ in terms of the definition of a truly detected text region. This is due to the fact that the text detection problem is fuzzier than just a simple true or false, where the overlap region between the detected and ground truth regions determines whether a text region has been detected or not. Therefore, the Intersection over Union (IoU) metric is defined as a ratio between 0 and 1 to determine the percentage at which the detected text region, or text regions, overlap with the ground truth text region:

$$IoU = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \tag{5.3}$$

While the precision P and recall R are defined as follows:

$$P = \frac{|TP|}{|D|}, \quad R = \frac{|TP|}{|G|} \tag{5.4}$$

where G and D are the sets of ground truth and detected text regions, respectively. As for the definition of TP, that is were different approaches have been taken. In this chapter two main performance measures were adopted to assess the performance of the various models on the DeTEXT test set:

- **The ICDAR 2003** [95] performance measure, which was used to evaluate the different methods submitted to the International Conference on Document Analysis and

Recognition (ICDAR) 2003. This metric matches each text region from one of the two sets with its best match from the other set, giving a different definition for TP between precision P and recall R:

$$\begin{aligned} TP_P(G,D) &= \sum_{i=1}^{|G|} BestMatch_G(G_i,D) \\ TP_R(G,D) &= \sum_{i=1}^{|D|} BestMatch_D(D_i,G) \end{aligned} \tag{5.5}$$

$$\begin{aligned} BestMatch_G(G_i,D) &= \max_{j=1\ldots|D|} \frac{2\times Area(G_i\cap D_j)}{Area(G_i)+Area(D_j)} \\ BestMatch_G(D_i,G) &= \max_{j=1\ldots|G|} \frac{2\times Area(D_i\cap G_j)}{Area(D_i)+Area(G_j)} \end{aligned} \tag{5.6}$$

In the ICDAR 2003 metric, only one-to-one matches were considered in the $BestMatch$ functions, which might disadvantage some methods that are capable of making a more precise predication. For instance, a ground truth text region could contain an entire sentence, but the detection method was able to generate more precise text regions that would surround each word in the sentence.

- **DetEval** [96] on the other hand, considers not only one-to-one matches but also one-to-many (splits) and many-to-one (merges) matches.

$$\begin{aligned} TP_P(G,D,t_p) &= \sum_i Match_D(D_i,G,t_p) \\ TP_R(G,D,t_r) &= \sum_i Match_G(G_i,D,t_r) \end{aligned} \tag{5.7}$$

$$Match_G(G_i, D, t_r) = \begin{cases} 0, & \text{if } G_i \text{ does not match any text region from } D \\ 1, & \text{if } G_i \text{ matches a single text region from } D \\ f_{sc}(k), & \text{if } G_i \text{ matches with } k \text{ number of text regions from } D \end{cases} \tag{5.8}$$

$$Match_D(D_i, G, t_p) = \begin{cases} 0, & \text{if } D_i \text{ does not match any text region from } G \\ 1, & \text{if } D_i \text{ matches a single text region from } G \\ f_{sc}(k), & \text{if } D_i \text{ matches with } k \text{ number of text regions from } G \end{cases} \tag{5.9}$$

$f_{sc}(k)$ being a parameter function that controls the punishment for splits and merges. If it is set to 1, no punishment is applied, and the lower the value the more the punishment. The DetEval results in this chapter have all been assessed with $f_{sc}(k) = 0.8$.

- **COCO** [86] This evaluation metric is borrowed from object detection and is used in this chapter to evaluate the performance of the proposed text localisation methods. The COCO evaluation metric is closer to the ICDAR one, where only the best matched region is considered to be the correct one. However, the COCO metric takes the average measures over a range of IoU thresholds. In this work, the average performance was taken over two IoU thresholds $\{0.5, 0.8\}$, which provides a middle ground between the 0.5 and 0.8 thresholds.

Using the above-mentioned metrics, a better insight can be acquired by comparing the three performance measures. Particularly because the DetEval metric considers splits and merges, while the other two measures do not, it is possible to get an idea about the degree of splits and merges produced by the predictor. Unfortunately, even though the COCO evaluation API supports the evaluation of oriented text regions, due to the ground truths of the DETEXT dataset, an accurate assessment of this prediction was not possible (more details in Section 5.7.3).

Table 5.1 Overall results achieved by the proposed model on the DETEXT test set [92].

Metric	Recall	Precision	F-measure
COCO	0.84	0.52	0.64
ICDAR2003	0.8	0.57	0.66
DetEval	0.68	0.48	0.56

5.6 Results

Using the test set from DETEXT, this section captures the performance of the proposed text localisation model. The model achieves an overall performance of 84% recall and 52% precision when assessed using the COCO performance measure. Table 5.1 shows the recall, precision and their harmonic mean (F-measure) when the model's output was evaluated using the three main metrics discussed in Section 5.5. The fact that evaluation metrics based on best match regions (ICDAR and COCO) show a higher performance than DetEval, implies that the proposed model is not generating many splits and merges that should be punished less by DetEval's parameter function $f_{sc}(k)$.

This section also breaks down the performance measure on the different challenge-categories discussed in Section 5.3. After acquiring the overall recall and precision for the model using the three different performance metrics, the recall for each challenge-category was calculated. This was possible only for recall and not for precision, because unlike classification or object detection tasks that include the prediction of the object class, the model was not classifying the type of text region. However, it is beneficial to calculate the recall for each challenge-category to understand the strengths and weaknesses of the model. In this case the classes specific recall (R_c) would be:

$$R_c = \frac{|TP_c|}{|G_c|} \tag{5.10}$$

where TP_c is the set of text regions that were correctly detected from the challenge-category c, and G_c is the set of ground truth text regions from the c challenge-category. Calculating the class specific precision on the other hand would not be possible, because the set D_c of text regions detected as the challenge-category c does not exist, as the model is not classifying the text regions. Table 5.2 shows the recall for each challenge-category c from the DETEXT test set. Following that, Table 5.3 shows the recall for the different combinations

Table 5.2 Performance breakdown on the different text regions categories from the DeTEXT dataset.

Category	# Regions		# Figures		Recall
	Training	Testing	Training	Testing	
Normal	1328	2191	159	256	0.87
Small	1186	1233	73	78	0.92
Blurry	646	472	29	36	0.65
Colour	63	230	10	29	0.84
Short	1744	2610	144	235	0.9
Complex Background	396	294	39	47	0.8
Complex Symbol	132	128	38	42	0.78
Specific Text	18	56	7	7	0.84

of challenge categories from the test set. When calculating the recall for the combination challenge-categories, each unique set of challenge-categories is treated as a single category c and the recall is calculated using Equation 5.10.

5.7 Discussion

5.7.1 Strengths

As shown in Table 5.2, the proposed model was able to achieve state-of-the-art recall rates on the different types of challenges. This proves the model's ability in dealing with the challenges posed by the nature of text regions in biomedical figures. To focus on those challenges, the model did not have to compromise on the recall for the normal text regions, where the model achieved a recall of 0.87.

The biggest two challenges faced when detecting text in biomedical figures are small and short text regions. Transforming the localisation problem into a reconstruction one, which operates on pixel level, allows for better detection of those types of text regions as demonstrated by the results in Table 5.2. Other models such as YOLO [73] and the different variations of the Region-based Convolutional Network (R-CNN) model [97, 98] try to classify images at a grid level, which is more suitable for object detection with larger objects.

Table 5.3 Performance breakdown on the different combined text region categories from the DeTEXT dataset.

Category	**# Regions**	**# Figures**	**Recall**
Small and short	1786	126	0.85
Small and blurry	858	47	0.83
Small and colour	28	7	1.0
Small and complex background	106	13	0.33
Small and complex symbol	19	9	0.00
Small, colour and complex background	15	2	0.86
Small, blurry and short	485	33	0.65
Small, short and complex background	47	8	1.0
Small, colour and short	10	4	0.89
Small, blurry and complex symbol	7	5	1.0
Small, blurry and complex background	43	4	0.17
Blurry and short	603	44	0.9
Short and complex background	279	48	0.5
Short and colour	96	22	0.75
Short and complex symbol	71	18	0.77
Colour and complex symbol	2	1	1.0
Colour and specific text	35	2	0.11
Colour and complex background	81	16	0.89
Colour, short and complex background	24	9	0.5
Complex background and specific text	3	2	0.0
Complex background and complex symbol	23	9	1.0

Through pre-training and data augmentation, the model was able to achieve high recall on text regions with complex backgrounds and those with coloured text. Considering the fact that the training data only had 52 coloured text regions, the SythText in the wild dataset and the colour inverse function that was implemented were able to introduce sufficient variability for the model to cope with the test data. The pre-training stage was also able to enhance the model's ability to deal with complex backgrounds, where the text in the SynthText dataset was generated over complex images.

5.7.2 Noise

Figure 5.6 shows sample outputs of the different challenge-categories. It is clear from Figure 5.6c that the model was struggling with blurry text, which is also clear from the results in Table 5.2. This was likely due to the lack of blurry text in the pre-training stage, where the synthetically generated texts are all quite clear. In the future, a function that introduces blurry text to the image, or perhaps blurs the entire image, could enhance the performance on this category. Looking at the same Figure 5.6c, some noise is visible in the place of some portions of the circles from the original figure. Such noise is also visible in place of some of the dashes in Figure 5.6a. This noise was mostly caused by shapes similar to text characters such as dashes (Figure 5.6a), circles and rectangles (Figure 5.7). Circles are particularity challenging because of their visual resemblance to the number zero "0" and to the letter "O". Making this a bigger challenge, the occurrence of such shapes is limited in the training data and does not exist in the pre-training dataset.

Complex symbols posed a significant challenge for the model, where the model achieved the second lowest recall. This was particularly challenging when the complex symbols were also in small font, making the recall for small text regions containing complex symbols 0 (Table 5.3).

Even though the model was able to achieve a recall of 0.84 on text regions with specific text, it performed poorly when specific text was coloured or over a complex background. This is because the model's ability to deal with coloured and complex backgrounds was mostly gained from the pre-training stage, where the SynthText dataset did not contain any specific text.

Table 5.4 Performance comparison between horizontal and oriented text regions from the DeTEXT dataset.

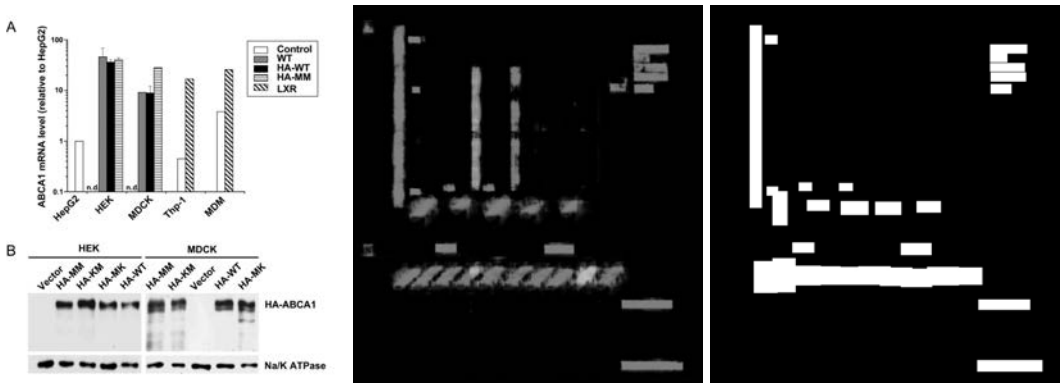
Category	**# Regions**	**# Figures**	**Recall**
Horizontal	8461	492	0.87
Oriented	847	268	0.66

5.7.3 Oriented Text Regions

In Table 5.4, the performance of the proposed model is compared between horizontal and oriented text regions. The model showed a better recall on horizontal text regions, which is the more represented class in this case. As mentioned earlier in Section 5.3, oriented text regions account for over 9% of the total text regions in the DeTEXT dataset. Therefore, random rotation was one of the data augmentation techniques that was implemented to counter the effect of this imbalance. Surprisingly, the random rotations and the pre-training using the synthetic dataset was not sufficient to make the model identify oriented texts as well as horizontal ones. When comparing this result with the performance of the model on the different categories of text regions in Table 5.2, it is apparent that model's performance is close to that for blurry text regions.

Figure 5.5 helps to explain the comparably low performance of the model on oriented text regions, especially when looking at the ground truth in Figure 5.5c. It becomes clear that the issue was with the dataset ground truths that depend on four points to define a text region. From Figure 5.5b, more noise is visible surrounding the oriented text regions, which is caused by the low confidence of the model in those areas being text regions.

This caused by a conflict between the training on the SynthText in the Wild dataset and that done using the DeTEXT dataset. Using the ground truths from the SynthText in the Wild dataset, the model was able to learn to detect oriented texts. However, in the next training stage using the DeTEXT dataset, this skill was overridden. This issue also explains more about the overall low precision achieved by the model, where the oriented text regions detected by our proposed model are counted as false positives.



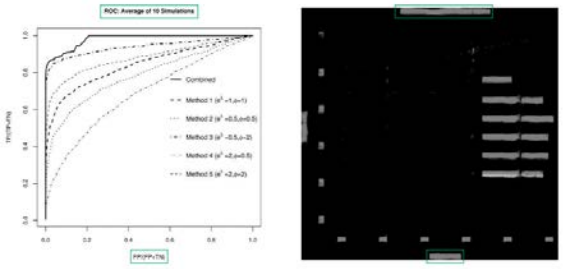
(a) Original (b) Reconstruction (c) Ground truth

Fig. 5.5 A figure containing oriented text regions.

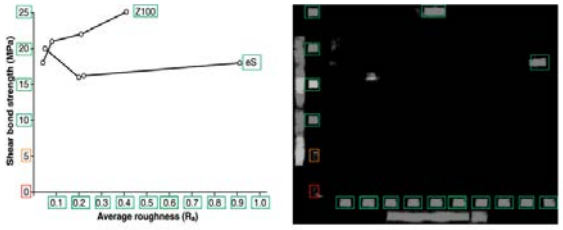
5.8 Conclusions

Localising text from biomedical figures images was addressed in this chapter, where a novel deep convolutional neural network was proposed. The deep model was designed and trained to resolve some of the specific challenges that are faced when localising text from biomedical figures. The novel architecture allowed for the simplification of the text extraction problem into a reconstruction one, where the output of the model is a black image with white regions where text is predicted to be. The chapter then discussed a pre-training stage for the model on a huge synthetic dataset of natural scene images. Different data augmentation techniques are then introduced to improve the prediction on specific categories of text regions. The chapter then delves into the results and in-depth analysis of the results achieved by the model, along with the strengths and weaknesses.

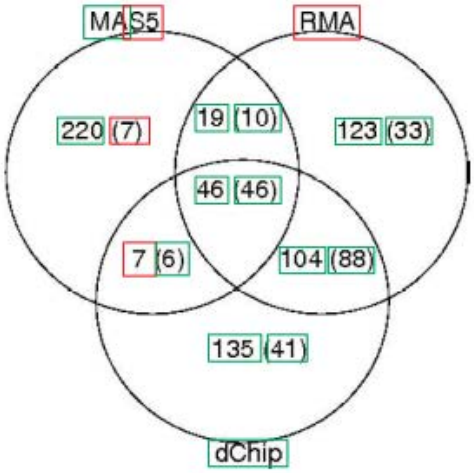
The outputs of the model proposed in this chapter would be used towards the end-to-end extraction of text not only from biomedical figures, but other contexts as well. Additionally, the reconstructed images could potentially be put as another feature towards the classification of biomedical figures, adding upon the work done in Chapter 4. Such feature could improve upon the classification performance, which would make the method proposed in this chapter a method for supervised representation learning. Furthermore, the text that could be easily extracted using an off-the-shelf OCR tool, such as the work done by Ma et al. [19], could be used as further indexing terms for figures beyond the figure class from Chapter 4. Such text could also be used to classify biomedical figures as an addition to visual features extracted using methods developed in Chapter 4.



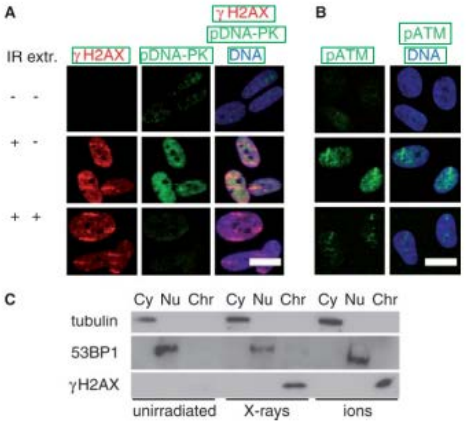
(a) Normal text



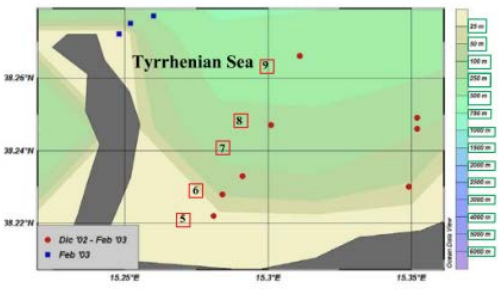
(b) Short text

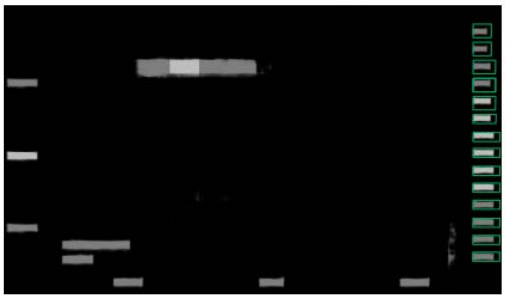


(c) Blurry text



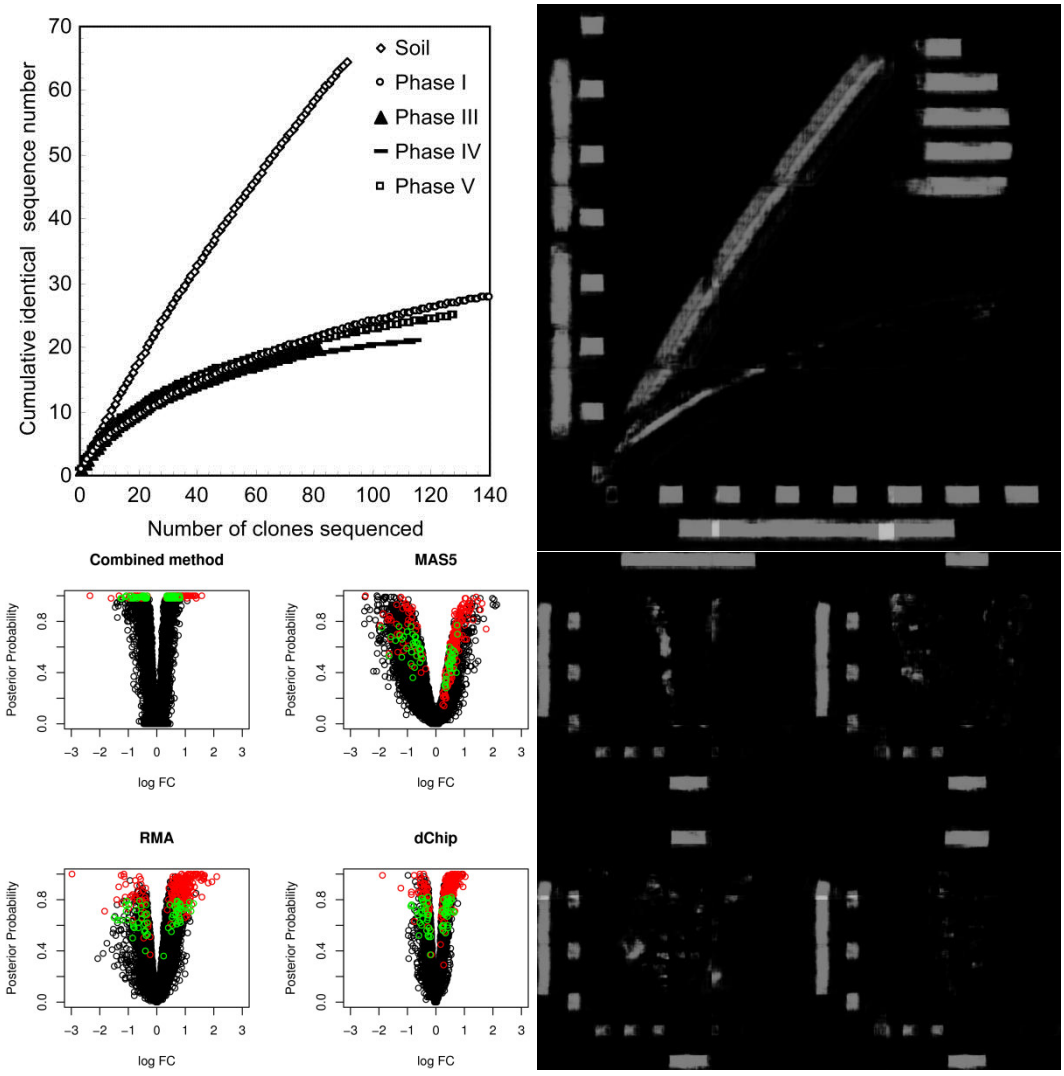
(d) Colour text





(e) Small text

Fig. 5.6 Examples of figures containing the main text localisation challenges.



(a) Input figures (b) Output reconstructions

Fig. 5.7 An example of chart plots with high impact from patterns similar to text.

Chapter 6

Conclusions

6.1 Thesis Contributions

The aim of this thesis is to develop various deep learning methods to pave the way for the effective extraction of information from figures in biomedical literature. It offers original approaches that add to the scientific works within the deep and representation learning areas. Through the use of those approaches, the thesis offers new ways of extracting information from biomedical figures, and thus contributing to the biomedical imaging field in general.

The contributions of this thesis are distributed across Chapters 3, 4 and 5. Those contributions can be summarised as follows:

- An effective representation learning training method that is aimed towards classification tasks in general, and towards the biomedical figure classification task in particular.
- A stacked deep autoencoder model for the automatic extraction of visual features from biomedical figures.
- An effective hierarchical ensemble of SVMs for biomedical figure classification.
- Deep convolutional models for biomedical figure classification, trained using our state-of-the-art representation learning method.
- A novel deep convolutional model to simplify the text localisation task into a reconstruction one.

The thesis offered comprehensive experimentation and discussion in support of the above-mentioned contributions. The novelty of this work is also supported through the following aspects: a) Identifying a link between the class separation ratio and the SVM cost value. b) Devising a multi-stage fine-tuning process to tackle the data class imbalance. c) Implementing different pre-training and data augmentation techniques to tackle the text localisation challenges in biomedical figures. The next sections of this chapter will shed light on the specific conclusions of each of the chapters and how they relate to the contributions mentioned above. Following from the introduction chapter, which provides the context for the thesis, comes the following chapters:

Chapter 2 – *Background and literature review* explores the backgrounds and the necessary knowledge behind the main methods that were developed in the later chapter of the thesis. The chapter also provides an idea of some of the work that has been previously done towards: 1) the classification of biomedical figures 2) text localisation from biomedical figures.

Chapter 3 – *Effective representation learning.* From the motivation to automatically extract features from biomedical figures, the chapter sought to establish a method to separate classes in a latent feature space, while also reducing the scatter within the classes themselves. The chapter details the different aspects of the method as well as the experimental settings followed to provide it with a solid ground. The method, called gradual class separation (GCS), is tested on two popular benchmark datasets and compared to the ubiquitous classification training techniques. Meanwhile, a link was established between the separation ratio and the SVM cost, which changed the SVM cost from a constant set at the beginning of an experiment to a variable that fluctuates depending on the current separation state of the classes in the latent feature space. An in-depth analysis was then carried out assessing the method's generalisation ability, accuracy and computational cost. In this chapter, GCS was tested on perfectly balanced datasets, CIFAR-10 and CIFAR-100, making the task of the next chapter to test the method's ability in dealing with (class) unbalanced datasets and the classification of biomedical figures.

Chapter 4 – *Figure classification* presents several methods for the classification of figures in biomedical literature, which is the first piece of information that could be extracted from figures' visual features. However, before delving into the methodology, the chapter

starts with an introduction to the dataset and taxonomies that are used, as well as the specific challenges that are faced when it comes to the classification of biomedical figures. Following that, the first method is introduced, which is a stacked deep autoencoder for the automatic extraction of features from biomedical figures [88]. Then, the training of deep residual neural networks for biomedical figure classification is described using the proposed method, GCS, and compared to the same architecture trained using cross-entropy and stochastic gradient descent. The chapter then looks at detailed results from the different models on the dataset of biomedical figures. Furthermore, the chapter analyses the performance difference between the models, with an in-depth discussion regarding the class imbalance problem and its effect on the proposed methods. The models described in this chapter gradually improve the classification accuracy of biomedical figures, while introducing the GCS method allows for the training of deeper models in faster times.

Chapter 5 – *Text localisation*. The work presented in this chapter is, to the best of our knowledge, the first deep learning effort into the localisation of text in biomedical figure images. In a similar manner to the previous chapter, it starts with an introduction to the adopted dataset, along with identifying the challenges that make the text localisation task from biomedical figures a unique task. Afterwards, a model inspired by autoencoders was proposed, where the localisation task was simplified into a reconstruction one. A pre-training stage is also introduced using a very large dataset of synthetic text in natural scene images to further improve the model's generalisation ability. Additionally, a novel data augmentation strategy is introduced to target the specific challenges of text extraction from biomedical figures. In addition to the data augmentation strategy, a novel splitting and stitching technique is used to enhance the model's performance on smaller texts. The performance of the model is then broken down using a variety of well-established metrics for text localisation. This chapter paves the way for the use of an off-the-shelf OCR tool to extract text from biomedical figures to be used towards indexing the figures and the articles they belong to.

6.2 Future Prospects and Research Constraints

Finding ways to identify an "optimal number of dimensions" to represent some input data is an interesting and promising problem that was encountered during the work on Chapter 3.

Another area that Chapter 3 could lead to is semi-supervised learning, where unlabelled data could be introduced as well as labelled ones into every input batch.

The biggest constraint that has faced this research was the shortage of labelled data, even though there is an enormous number of open-access biomedical articles available. This lack of labelled data had an effect on both the figure classification and text localisation tasks. Semi-supervised learning would also be beneficial for the figure classification task as a way of dealing with the small amount of labelled data available, especially for the smaller classes within the taxonomy.

As for text extraction, training a model for the localisation and extraction of text simultaneously would be an avenue to explore with biomedical figures. For the time being, this is restricted by the availability of character level annotation of a biomedical figure dataset. Furthermore, formulating a biomedical text-figure dataset with text region definitions that support oriented texts would help with improving the performance of any developed model and it would also help with the assessment of its true performance.

Text correction is another avenue to be developed following the extraction of text from figures. Such models could be more dependent on the context of the paper or even the figure, using specialised biomedical lexicons. However, this would require a dataset that links each figure with its source paper to provide the necessary context. This context could also be assisted by the classification task that determines the nature of the figure containing the text.

Finally, this thesis has been a great journey, through which I have found myself in the field of representation learning. I sincerely hope that the work done in this thesis will provide useful grounds for upcoming work on the extraction of information from biomedical figures.

References

[1] Oxford University Press, "Figure | Definition of figure in English by Oxford Dictionaries." https://en.oxforddictionaries.com/definition/figure.

[2] C. Clark and S. Divvala, "Looking beyond text: Extracting figures, tables and captions from computer science papers," in *AAAI 2015 Workshop on Scholarly Big Data*, 2015.

[3] L. D. Lopez, J. Yu, C. N. Arighi, H. Huang, H. Shatkay, and C. Wu, "An Automatic System for Extracting Figures and Captions in Biomedical PDF Documents," pp. 578–581, IEEE, Nov. 2011.

[4] S. R. Choudhury, P. Mitra, A. Kirk, S. Szep, D. Pellegrino, S. Jones, and C. L. Giles, "Figure Metadata Extraction from Digital Documents," in *2013 12th International Conference on Document Analysis and Recognition*, pp. 135–139, Aug. 2013.

[5] M. Taschwer and O. Marques, "Compound Figure Separation Combining Edge and Band Separator Detection," in *MultiMedia Modeling* (Q. Tian, N. Sebe, G.-J. Qi, B. Huet, R. Hong, and X. Liu, eds.), vol. 9516, pp. 162–173, Cham: Springer International Publishing, 2016.

[6] Apache, "Apache PDFBox | A Java PDF Library." https://pdfbox.apache.org/.

[7] freedesktop.org, "Poppler." https://poppler.freedesktop.org/.

[8] Glyph & Cog, "XpdfReader." https://www.xpdfreader.com/.

[9] R. F. Murphy, M. Velliste, J. Yao, and G. Porreca, "Searching online journals for fluorescence microscope images depicting protein subcellular location patterns," in *Bioinformatics and Bioengineering Conference, 2001. Proceedings of the IEEE 2nd International Symposium On*, pp. 119–128, IEEE, 2001.

[10] S. Antani, D. Demner-Fushman, J. Li, B. V. Srinivasan, and G. R. Thoma, "Exploring use of images in clinical articles for decision support in evidence-based medicine," in *Electronic Imaging 2008* (B. A. Yanikoglu and K. Berkner, eds.), (San Jose, CA), pp. 68150Q–68150Q–10, Jan. 2008.

[11] L. D. Lopez, J. Yu, C. Arighi, C. O. Tudor, M. Torii, H. Huang, K. Vijay-Shanker, and C. Wu, "A framework for biomedical figure segmentation towards image-based document retrieval," *BMC Systems Biology*, vol. 7, p. S8, Oct. 2013.

[12] L. T. Lam, O. K. Pickeral, A. C. Peng, A. Rosenwald, E. M. Hurt, J. M. Giltnane, L. M. Averett, H. Zhao, R. E. Davis, M. Sathyamoorthy, L. M. Wahl, E. D. Harris, J. A. Mikovits, A. P. Monks, M. G. Hollingshead, E. A. Sausville, and L. M. Staudt, "Genomic-scale measurement of mRNA turnover and the mechanisms of action of the anti-cancer drug flavopiridol," *Genome Biology*, vol. 2, p. research0041.1, Sept. 2001.

[13] S. Xu, J. McCusker, and M. Krauthammer, "Yale Image Finder (YIF): A new search engine for retrieving biomedical images," *Bioinformatics*, vol. 24, pp. 1968–1970, Sept. 2008.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, (Santiago, Chile), pp. 1026–1034, IEEE, Dec. 2015.

[15] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of Go without human knowledge," *Nature*, vol. 550, pp. 354–359, Oct. 2017.

[16] V. S. N. Prasad, B. Siddiquie, J. Golbeck, and L. S. Davis, "Classifying Computer Generated Charts," in *2007 International Workshop on Content-Based Multimedia Indexing*, pp. 85–92, June 2007.

[17] H. Shatkay, N. Chen, and D. Blostein, "Integrating image data into biomedical text categorization," *Bioinformatics*, vol. 22, pp. e446–e453, July 2006.

[18] A. G. S. de Herrera, D. Markonis, and H. Müller, "Bag–of–colors for biomedical document image classification," in *Medical Content-Based Retrieval for Clinical Decision Support*, pp. 110–121, Springer, 2012.

[19] K. Ma, H. Jeong, M. V. Rohith, G. Somanath, R. Tarpine, K. Schutter, D. Blostein, S. Istrail, C. Kambhamettu, and H. Shatkay, "Utilizing image-based features in biomedical document classification," in *Image Processing (ICIP), 2015 IEEE International Conference On*, pp. 4451–4455, IEEE, 2015.

[20] B. Rafkind, M. Lee, S.-F. Chang, and H. Yu, "Exploring text and image features to classify images in bioscience literature," in *Proceedings of the Workshop on Linking Natural Language Processing and Biology: Towards Deeper Biological Literature Analysis*, pp. 73–80, Association for Computational Linguistics, 2006.

[21] X.-H. Han and Y.-W. Chen, "Biomedical Imaging Modality Classification Using Combined Visual Features and Textual Terms," *International Journal of Biomedical Imaging*, vol. 2011, pp. 1–7, 2011.

[22] D. Kim, B. P. Ramesh, and H. Yu, "Automatic figure classification in bioscience literature," *Journal of Biomedical Informatics*, vol. 44, pp. 848–858, Oct. 2011.

[23] Y. Gkoufas, A. Morou, and T. Kalamboukis, "Combining textual and visual information for image retrieval in the medical domain," *The Open Medical Informatics Journal*, vol. 5, pp. 50–57, 2011.

[24] M. S. Simpson, D. You, M. M. Rahman, Z. Xue, D. Demner-Fushman, S. Antani, and G. Thoma, "Literature-based biomedical image classification and retrieval," *Computerized Medical Imaging and Graphics*, vol. 39, pp. 3–13, Jan. 2015.

[25] J. Zhang, Y. Xia, Q. Wu, and Y. Xie, "Classification of Medical Images and Illustrations in the Biomedical Literature Using Synergic Deep Learning," *arXiv preprint arXiv:1706.09092*, 2017.

[26] A. Kumar, J. Kim, D. Lyndon, M. Fulham, and D. Feng, "An Ensemble of Fine-Tuned Convolutional Neural Networks for Medical Image Classification," *IEEE Journal of Biomedical and Health Informatics*, vol. 21, pp. 31–40, Jan. 2017.

[27] L. D. Lopez, J. Yu, C. N. Arighi, M. Torii, K. Vijay-Shanker, H. Huang, and C. H. Wu, "An Image-Text Approach for Extracting Experimental Evidence of Protein-Protein Interactions in the Biomedical Literature," in *Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*, p. 412, ACM, 2013.

[28] H. Müller, J. Kalpathy-Cramer, D. Demner-Fushman, and S. Antani, "Creating a classification of image types in the medical literature for visual categorization," in *SPIE Medical Imaging* (W. W. Boonn and B. J. Liu, eds.), vol. 8319, Feb. 2012.

[29] A. G. S. de Herrera, J. Kalpathy-Cramer, D. Demner-Fushman, S. K. Antani, and H. Müller, "Overview of the ImageCLEF 2013 Medical Tasks.," in *CLEF (Working Notes)*, 2013.

[30] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[31] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[32] A. Krizhevsky and G. E. Hinton, "Using very deep autoencoders for content-based image retrieval.," in *ESANN*, vol. 1, p. 2, Citeseer, 2011.

[33] J. Geng, J. Fan, H. Wang, X. Ma, B. Li, and F. Chen, "High-Resolution SAR Image Classification via Deep Convolutional Autoencoders," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, pp. 2351–2355, Nov. 2015. Conference Name: IEEE Geoscience and Remote Sensing Letters.

[34] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. MIT press Cambridge, 2016.

[35] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv:1409.1556 [cs]*, Sept. 2014.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385 [cs]*, Dec. 2015.

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, (Miami, FL), pp. 248–255, IEEE, June 2009.

[38] M. A. Nielsen, *Neural Networks and Deep Learning*. 2015.

[39] S. Koitka and C. M. Friedrich, "Traditional Feature Engineering and Deep Learning Approaches at Medical Classification Task of ImageCLEF 2016.," in *CLEF (Working Notes)*, pp. 304–317, 2016.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[42] A. Koike and T. Takagi, "Classifying Biomedical Figures Using Combination of Bag of Keypoints and Bag of Words," pp. 848–853, IEEE, Mar. 2009.

[43] R. Rodriguez-Esteban and I. Iossifov, "Figure mining for biomedical research," *Bioinformatics*, vol. 25, pp. 2082–2084, Aug. 2009.

[44] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, Sept. 1995.

[45] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 1–27, Apr. 2011.

[46] Chih-Wei Hsu and Chih-Jen Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, pp. 415–425, Mar. 2002.

[47] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene Selection for Cancer Classification using Support Vector Machines," *Machine Learning*, vol. 46, pp. 389–422, Jan. 2002.

[48] Abdul Rahim Ahmad, M. Khalia, C. Viard-Gaudin, and E. Poisson, "Online handwriting recognition using support vector machine," in *2004 IEEE Region 10 Conference TENCON 2004.*, vol. A, pp. 311–314 Vol. 1, Nov. 2004.

[49] E. Osuna, R. Freund, and F. Girosit, "Training support vector machines: An application to face detection," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (San Juan, Puerto Rico), pp. 130–136, IEEE Comput. Soc, 1997.

[50] R. A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems," *Annals of Eugenics*, vol. 7, pp. 179–188, Sept. 1936.

[51] M. Dorfer, R. Kelz, and G. Widmer, "Deep Linear Discriminant Analysis," *arXiv:1511.04707 [cs]*, Nov. 2015.

[52] A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature Extraction With Deep Neural Networks by a Generalized Discriminant Analysis," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, pp. 596–608, Apr. 2012.

[53] E. Osuna, R. Freund, and F. Girosi, "Support Vector Machines: Training and Applications," tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA, 1997.

[54] J.-H. Xue and D. M. Titterington, "Do unbalanced data have a negative effect on LDA?," *Pattern Recognition*, vol. 41, pp. 1558–1571, May 2008.

[55] J. Xie and Z. Qiu, "The effect of imbalanced data sets on LDA: A theoretical and empirical analysis," *Pattern Recognition*, vol. 40, pp. 557–562, Feb. 2007.

[56] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie, "COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images," *arXiv:1601.07140 [cs]*, Jan. 2016.

[57] A. Antonacopoulos, D. Bridson, C. Papadopoulos, and S. Pletschacher, "A Realistic Dataset for Performance Evaluation of Document Layout Analysis," in *2009 10th International Conference on Document Analysis and Recognition*, (Barcelona, Spain), pp. 296–300, IEEE, 2009.

[58] X.-C. Yin, C. Yang, W.-Y. Pei, H. Man, J. Zhang, E. Learned-Miller, and H. Yu, "DeTEXT: A Database for Evaluating Text Extraction from Biomedical Literature Figures," *PLOS ONE*, vol. 10, p. e0126200, May 2015.

[59] S. Xu and M. Krauthammer, "A new pivoting and iterative text detection algorithm for biomedical images," *Journal of Biomedical Informatics*, vol. 43, pp. 924–931, Dec. 2010.

[60] S. Xu and M. Krauthammer, "Boosting text extraction from biomedical images using text region detection," in *Biomedical Sciences and Engineering Conference (BSEC), 2011*, pp. 1–4, IEEE, 2011.

[61] D. Kim and H. Yu, "Figure Text Extraction in Biomedical Literature," *PLoS ONE*, vol. 6, p. e15338, Jan. 2011.

[62] B. Gatos, I. Pratikakis, and S. Perantonis, "Text Detection in Indoor/Outdoor Scene Images," in *Proceedings of the 1st International Workshop on Camera-Based Document Analysis and Recognition, CBDAR 2005*, Jan. 2005.

[63] Kwang In Kim, Keechul Jung, and Jin Hyung Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1631–1639, Dec. 2003.

[64] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *In Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 366–373, 2004.

[65] A. Mishra, K. Alahari, and C. V. Jawahar, "Top-down and bottom-up cues for scene text recognition," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, (Providence, RI), pp. 2687–2694, IEEE, June 2012.

[66] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (San Francisco, CA, USA), pp. 2963–2970, IEEE, June 2010.

[67] C. Yi and Y. Tian, "Text String Detection from Natural Scenes by Structure-based Partition and Grouping," *Ieee Transactions on Image Processing*, vol. 20, pp. 2594–2605, Sept. 2011.

[68] H. I. Koo and D. H. Kim, "Scene Text Detection via Connected Component Clustering and Nontext Filtering," *IEEE Transactions on Image Processing*, vol. 22, pp. 2296–2305, June 2013.

[69] Yi-Feng Pan, Xinwen Hou, and Cheng-Lin Liu, "A Hybrid Approach to Detect and Localize Texts in Natural Scene Images," *IEEE Transactions on Image Processing*, vol. 20, pp. 800–813, Mar. 2011.

[70] L. Neumann and J. Matas, "Scene Text Localization and Recognition with Oriented Stroke Detection," in *2013 IEEE International Conference on Computer Vision*, (Sydney, Australia), pp. 97–104, IEEE, Dec. 2013.

[71] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-End Text Recognition with Convolutional Neural Networks," p. 5.

[72] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *arXiv:1311.2524 [cs]*, Nov. 2013.

[73] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Las Vegas, NV, USA), pp. 779–788, IEEE, June 2016.

[74] H. Li, T. Jiang, and K. Zhang, "Efficient and robust feature extraction by maximum margin criterion," in *Advances in Neural Information Processing Systems*, pp. 97–104, 2004.

[75] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. Mullers, "Fisher discriminant analysis with kernels," in *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*, (Madison, WI, USA), pp. 41–48, IEEE, 1999.

[76] M. H. Nguyen and F. de la Torre, "Optimal feature selection for support vector machines," *Pattern Recognition*, vol. 43, pp. 584–591, Mar. 2010.

[77] G. F. Elsayed, D. Krishnan, H. Mobahi, K. Regan, and S. Bengio, "Large Margin Deep Networks for Classification," *arXiv:1803.05598 [cs, stat]*, Mar. 2018.

[78] A. Krizhevsky, *Learning Multiple Layers of Features from Tiny Images*. PhD thesis.

[79] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *arXiv:1212.5701 [cs]*, Dec. 2012.

[80] T. Joachims, "Making large-scale SVM learning practical," Working Paper 1998,28, Technical Report, 1998.

[81] L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[82] H. Müller, A. García Seco de Herrera, and S. Bromuri, "Overview of the ImageCLEF 2015 medical classification task." https://hesso.tind.io/record/1025, 2015.

[83] Y. Bengio, "Learning Deep Architectures for AI," *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[84] G. E. Hinton, "To recognize shapes, first learn to generate images," in *Progress in Brain Research*, vol. 165, pp. 535–547, Elsevier, 2007.

[85] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of Machine Learning Research* (F. Bach and D. Blei, eds.), vol. 37, (Proceedings of Machine Learning Research), pp. 448–456, PMLR, 2015.

[86] H.-T. Lin, C.-J. Lin, and R. C. Weng, "A note on Platt's probabilistic outputs for support vector machines," *Machine Learning*, vol. 68, pp. 267–276, Aug. 2007.

[87] T.-f. Wu, C.-j. Lin, and R. C. Weng, "Probability Estimates for Multi-Class Classification by Pairwise Coupling," in *Advances in Neural Information Processing Systems 16* (S. Thrun, L. K. Saul, and B. Schölkopf, eds.), pp. 529–536, MIT Press, 2004.

[88] I. Almakky, V. Palade, Y. Hedley, and J. Yang, "A stacked deep autoencoder model for biomedical figure classification," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 1134–1138, Apr. 2018.

[89] G. King and L. Zeng, "Logistic Regression in Rare Events Data," SSRN Scholarly Paper ID 1083726, Social Science Research Network, Rochester, NY, 2001.

[90] C. Yang, X. Yin, H. Yu, D. Karatzas, and Y. Cao, "ICDAR2017 Robust Reading Challenge on Text Extraction from Biomedical Literature Figures (DeTEXT)," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 01, pp. 1444–1447, Nov. 2017.

[91] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic Data for Text Localisation in Natural Images," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Las Vegas, NV, USA), pp. 2315–2324, IEEE, June 2016.

[92] I. Almakky, V. Palade, and A. Ruiz-Garcia, "Deep Convolutional Neural Networks for Text Localisation in Biomedical Literature Figures," in *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–5, IEEE, July 2019.

[93] J. Duchi, E. Hazan, and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, July 2011.

[94] S. Suzuki and K. be, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, pp. 32–46, Apr. 1985.

[95] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, K. Ashida, H. Nagai, M. Okamoto, H. Yamamoto, H. Miyao, J. Zhu, W. Ou, C. Wolf, J.-m. Jolion, L. Todoran, M. Worring, and X. Lin, "ICDAR 2003 robust reading competitions: Entries, results and future directions," in *International Journal on Document Analysis and Recognition - Special Issue on Camera-Based Text and Document Recognition 7(2–3*, pp. 105–122, 2005.

[96] C. Wolf and J.-M. Jolion, "Object count/area graphs for the evaluation of object detection and segmentation algorithms," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 8, no. 4, pp. 280–296, 2006.

[97] X. Wang, A. Shrivastava, and A. Gupta, "A-Fast-RCNN: Hard Positive Generation via Adversary for Object Detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3039–3048, July 2017.

[98] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *arXiv:1703.06870 [cs]*, Mar. 2017.