Exploring Object Detection and Recognition Methods for Automated Book Inventory

Anna Svensson

Master's thesis
2019:E62

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

CENTRUM SCIENTIARUM MATHEMATICARUM

Master Thesis

Exploring Object Detection and Recognition Methods for Automated Book Inventory

December 9, 2019

LUND UNIVERSITY

by Anna Svensson

LU Supervisor: Karl Åström
LU Co-supervisor: Kenneth Batstone
Neodev Co-supervisor: Johan Sjögren
Neodev Co-supervisor: Daniel Ståhl
Examiner: Carl Olsson

Centre for Mathematical Sciences
Lund Institute of Technology, Lund University
Lund, Sweden 2019

Abstract

As more modern technical devices with cameras are accessible to people, more photos are taken and the amount of data that is available has vastly increased. Researchers have learnt to utilise this and use images for various purposes with Deep Learning based Computer Vision. For instance, one can teach a computer to perform an automated inventory, which could be a very time consuming task. Librarians are some of the affected people, they spend a lot of their time to arrange and go through the books on the bookshelves. In an attempt to automate the inventory of books, different object detection and recognition methods have been explored in this Master Thesis. The state of the art object detection method Mask Region Convolutional Neural Network (Mask R-CNN) was used to find book spines in images and two Convolutional Neural Networks (CNN) with different loss functions; categorical cross-entropy and triplet loss, were used to recognise each book the Mask R-CNN had found. In addition, various data augmentation techniques were used to artificially expand and diversify the training dataset. The Mask R-CNN reached a mean Average Precision (mAP) of 94% on one of the test sets and the recognition networks each got 90% accuracy on the books that had been found.

Keywords: Object Detection, Deep Learning, Automated Book Inventory, Computer Vision, Data Augmentation, Mask R-CNN, Triplet Loss

Acknowledgements

First I would like to thank my supervisors. Johan Sjögren and Daniel Ståhl have provided me with heaps of ideas and explained many topics that I did not understand. As Neodev employees they have been available most of the time over email conversations or at the office, where most of the project has been formed, to answer questions and discuss ideas. I would like to thank Karl Åström and Kenneth Batstone, my supervisors from the Faculty of Engineering at Lund University, for excellently guiding me throughout the process and giving me lots of inspiration and support along the way.

I also want to thank the librarians at the libraries where I was allowed to take photos of books, especially Mikael Abrahamsson at the Mathematics's Library who gave me a detailed explanation of librarians' tasks and how they organise the bookshelves.

A big thank you to all the fantastic people at Neodev's office who never have refused a moment to discuss ideas or chat over a cup of coffee. I am grateful for their welcoming attitude and I am looking forward to our future cooperation.

Last but not least, I am thankful for the constant support and encouragement from my friends and family.

Contents

Abbreviations and Acronyms **6**

1 Introduction **7**
1.1 Background 7
1.2 Aim and Purpose 7
1.3 Tools and Resources 8
1.4 Outline 8

2 Theory **10**
2.1 Machine Learning for Computer Vision 10
2.1.1 Feature Extraction and Detection 10
2.1.2 Supervised and Unsupervised Learning 11
2.1.3 Underfitting and Overfitting 11
2.1.4 Evaluation 12
2.1.4.1 Accuracy, Recall, Precision 12
2.1.4.2 Mean Average Precision, mAP 13
2.1.4.3 Intersection over Union, IoU 13
2.2 Convolutional Neural Network, CNN 13
2.2.1 Convolutional Layer 14
2.2.2 Pooling Layer 14
2.2.3 Fully Connected Layer 15
2.2.4 Activation Functions 16
2.2.5 Training a CNN 17
2.2.5.1 Loss Function 17
2.2.5.2 Optimiser 19
2.2.5.3 Transfer Learning 19
2.3 Object Detection Networks 20
2.3.1 Region-based Convolutional Neural Network, R-CNN 20
2.3.2 Spatial Pyramid Pooling Network, SPPnet 20
2.3.3 Fast Region Convolutional Neural Network, Fast R-CNN 20
2.3.4 Faster Region Convolutional Neural Network, Faster R-CNN 21
2.3.5 State of the Art Models 21
2.3.5.1 Mask Region Convolutional Neural Network, Mask R-CNN 21
2.3.5.2 You Only Look Once, YOLO 21
2.3.5.3 Single Shot Detector, SSD 22
2.3.5.4 RetinaNet 22

3 Methodology **23**
3.1 Data gathering 23
3.1.1 Data augmentation 23
3.2 Training Neural Networks 25
3.2.1 Mask R-CNN 25
3.2.2 CNN with Categorical Cross-entropy Loss 25
3.2.3 CNN with Triplet Loss 26
3.3 Evaluation and Benchmarking 27

4 Results **28**
4.1 Object Detection with Mask R-CNN 28
4.2 Classification with Categorical Cross-entropy Loss 29
4.3 Classification with Triplet Loss 30

5 Discussion **32**
5.1 Data gathering 32
5.2 Object Detection 32
5.3 Recognition 33
5.4 Sources of Errors 34
5.5 Future Work and Suggestions 34

6 Conclusion **36**

7 References **37**

8 Appendix **40**

Abbreviations and Acronyms

Some abbreviations are frequently used in this report, a reference list is provided below.

- CNN - Convolutional Neural Network
- CS - Cosine Similarity
- fn - false negative
- fp - false positive
- Fast R-CNN - Fast Region Convolutional Neural Network
- Faster R-CNN - Faster Region Convolutional Neural Network
- FCN - Fully Connected Network
- mAP - mean Average Precision
- Mask R-CNN - Mask Region Convolutional Neural Network
- OCR - Optical Character Recognition
- PCA - Principal Component Analysis
- R-CNN - Region-based Convolutional Neural Network
- ReLU - Rectified Linear Unit
- ROI - Regions of Interest
- RPN - Region Proposal Network
- SGD - Stochastic Gradient Descent
- SPPnet - Spatial Pyramid Pooling network
- SSD - Single Shot Detector
- tn - true negative
- tp - true positive
- t-SNE - t-distributed Stochastic Neighbour Embedding
- VGG - Visual Geometry Group
- YOLO - You Only Look Once

1 Introduction

"Can Machines Think?" - Alan Turing

Ever since the concept of Artificial Intelligence was introduced we have asked ourselves if computers can imitate or even surpass human thinking. Scientists have tried to create artificial neural networks that work like the neurons in our brains, with the belief that a machine can think just like a human. During the last couple of decades, research in these areas combined with image analysis and computer vision has flourished and its outcome has been established in our every day lives. When we post an image on Facebook we get suggestions of tags for each face in the image. A lot of smartphones have fingerprint readers to unlock the them. Some new cars use pedestrian detection systems to prevent accidents and some have sensors that read the lines painted on the road so that it can drive on autopilot. These are just a few Machine Learning applications using images that are widely used today.

One of the challenges in Machine Learning with image analysis is when dealing with multiple objects in one image. In that case there are a few extra steps in which the image needs to be segmented into foreground and background, or into a number of objects. If the objects are close to each other it may be hard to say where one ends and another begins. An example of such objects are books in bookshelves.

Essential tasks for librarians are to inventory books, moving misplaced books in bookshelves and keeping them in order. Though the tasks can be excessively time consuming, it is important to perform them so that books can be found easily. This raises the idea of developing a system that can do this automatically, so that librarians can spend their time on more rewarding tasks, like helping people find the books they are looking for and encourage reading.

In this master thesis we will explore the usage of neural networks to recognise books from images of book spines and how to compare them with similar images. We will analyse what kind of data and how much data is needed to do such a book recognition. In the best case scenario this system could be used to perform an automated book inventory which gives us the title and author of every book in an image, given that the network has learnt the books, otherwise we want it to tell us if there are any new books in the image.

1.1 Background

Previous work on the inventory of books often focuses on Optical Character Recognition (OCR) to read the title and author of each image of a segmented book spine. Quoc and Choi [20] developed a framework for moving a camera into suitable positions for book recognition with OCR . A group at the Pennsylvania State University created a similar system but using the Hough Transform to extract the dominant direction of the book spines to rotate the image and then applying a text/non-text Convolutional Neural Network (CNN) model to read the text [32]. As OCR is not always successful due to noise, an American team created an inventory management system using feature-based book spine recognition instead [3]. Whereas Tsai et. al. [24] created a hybrid model, that combines image and text features which significantly increased the recall.

1.2 Aim and Purpose

The aim of the thesis is to explore ways to inventory books from images. To develop such a system, one possible way to register each book is using Deep Learning based Computer Vision. In this case a computer must learn to find the books in images and be able to distinguish each book, and this

is what will be analysed in this Master Thesis. Images of book spines will be used, as they would be seen in a library. The usage of different neural networks is to be examined, as well as different data augmentation methods.

Unlike previous work on book recognition, this approach focuses on Deep Learning methods. In [20], books were found with a line detector and they were recognised with OCR, much similar to [32]. In [3] and [24] they also used line detectors to find books but used a feature detector and descriptor to recognise the books, the latter used OCR as well. These are mainly Machine Learning methods, except for the CNN based OCR. In this thesis, the detection of books will be performed with one of the latest Deep Learning based object detection methods and the recognition will be done with CNN:s. The Machine Learning approaches do not require a lot of data which the Deep Learning approach in general benefits from, therefore one substantial part of this thesis is to gather much data.

1.3 Tools and Resources

To gather data, i.e. images of books, a Nikon D3100 camera with an AF-S DX NIKKOR 18-55mm F/3.5-5.6G VR lens was used. The photos were taken at the libraries *Lunds Stadsbibliotek*, *Matematiska biblioteket* at LTH, *Universitetsbiblioteket - Lunds Universitet* and the author's home.

The neural networks were trained and used on a *Lenovo ideapad 320S* with an Intel® Core™i5-8250U CPU primarily but as large datasets put demands on computational power, one needs a GPU or/and a lot of RAM to train these kind of networks. To train Mask R-CNN a more powerful computer than the author's personal was needed. This was solved using Google Cloud Platform (GCP) where amount of RAM and number of GPU:s and CPU:s could be adjusted as needed. The scripts were run on GCP's AI Platform. All code is written in Python and some of the libraries frequently used are scikit-learn, TensorFlow, Keras, OpenCV, Python Imaging Library (Pillow) and seaborn and Pandas for visualisations.

To generalise the set of images, book spines were cropped and pasted into background images. The background images were downloaded from *Unsplash* [30], a website with free stock photos. Data augmentation is further elaborated on in Section 3.1.

A few developers' code has been reused and modified to fit the book recognition problem. Ketaro Wada's implementation of *labelme* was used to annotate images [31]. The detection part was done with Waleed Abdulla's implementation of Mask R-CNN [1] and one of the classification methods was based on David Sandberg's FaceNet-implementation [25]. More information on these architectures can be found in Section 3.

Some pretrained weights were used to speed up the training of the neural networks. The weights used for object detection were pretrained COCO weights, which can be downloaded from Abdulla's Github repository. For classification, weights trained on the datasets CASIA-WebFace and VGGFace2 were used. They can be found at Sandberg's Github repository [25].

1.4 Outline

This project is divided into two parts, one where each book is to be found in an image, which requires object detection, and one where each book is recognised through classification. To get a grip on the subject, the theory Section is divided into three parts. Section 2.1 covers the theory behind Machine Learning with Computer Vision and some of the main topics within this area. The components of a Convolutional Neural Network and the training procedure will be briefly explained in Section 2.2 before we go through different Object Detection Networks in Section 2.3.

Section 3 briefs us on the procedure of gathering data for this specific problem and how the Neural Networks are trained and evaluated.

After that we will go through the results in Section 4 where one can read how well the Mask R-CNN detects books and how accurately a CNN can classify each detected book with two different loss functions.

Finally the results and chosen methods are discussed in Section 5 and a conclusion is given in Section 6.

In the appendix one can find visualisations of the triplet loss embeddings and a list of all the books that were photographed and labelled for the classification.

2 Theory

"Making a computer see was something that leading experts in the field of Artificial Intelligence thought to be at the level of difficulty of a summer student's project back in the sixties. Forty years later the task is still unsolved and seems formidable" - Olivier Faugeras

What we call Computer Vision began with researchers wanting to mimic the human visual system, fifty years ago. It is meant to automate extraction of useful information from images that can be used for many applications, for example to create 3D models, calculate camera position and perform object detection and recognition. Mostly the latter will be discussed in this thesis. Object detection and recognition may seem like a trivial problem as seeing is effortless for most people. It is not hard for us to recognise faces that we have seen or to tell where the boundaries of an object is. It is more problematic for computers and this is still a big research field today. A lot of progress has been made though. Modern Deep learning models can classify images of dogs and cats with 99% accuracy. Some object detection models are fast enough to use on real-time video and face recognition models can even outperform a person's ability to recognise [2].

A few central terms are good to know to understand the content, these are found below.

Epoch - One pass of the whole dataset.

Batch / Mini-batch - A part of the dataset that is passed though the network before calculating an error and updating the parameters. In literature, batch is sometimes used interchangeably as epoch or mini-batch. In this thesis, batch and mini-batch are used synonymously.

Weight - Weights are parameters connecting nodes in the network. These are trainable.

Learning rate - Defines how quickly the Neural Network updates the parameters. A high learning rate speeds up the training process but may prevent it from converging. A low slows it down but it usually converges, given enough time.

Hyper-parameters - Parameters that can be tuned but not trained by the network. Needs to be set before training, i.e. learning rate, number of epochs, batch size.

2.1 Machine Learning for Computer Vision

2.1.1 Feature Extraction and Detection

Feature extraction is an essential part in Machine Learning. Lots of different feature extraction and detection algorithms have been developed throughout the last decades. Some find lines, other find gradients or calculate local histograms and a few are even scale invariant. Though a lot of research has been made within feature engineering, one drawback with manual feature extraction is that it requires domain knowledge. If one wants to find zebras in an image, an algorithm that finds lines and gradients is a reasonable detector but if one wants to find elephants instead, the choice of an algorithm might not be as apparent. With Deep Learning however, one does not have to consider what kind of features are decisive, that is up to the deep neural network to determine. Deep Learning methods can find low level features as well as high level features, and it learns during training which features that are important. It can find important edges and gradients just like some of the algorithms outlined above but also essential shapes which lets us solve more complex problems. In Figure 1 the main difference in feature extraction using Machine Learning and Deep Learning is visualised and in Figure 2 one can see some examples of low, mid and high level features.

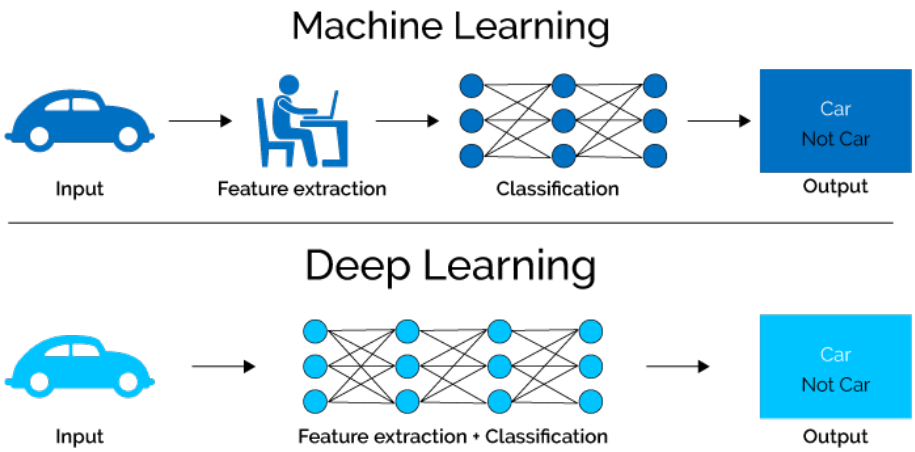

Figure 1: In Machine Learning one uses manual feature extraction while a Deep Learning network learns this extraction during training. Image from [18].

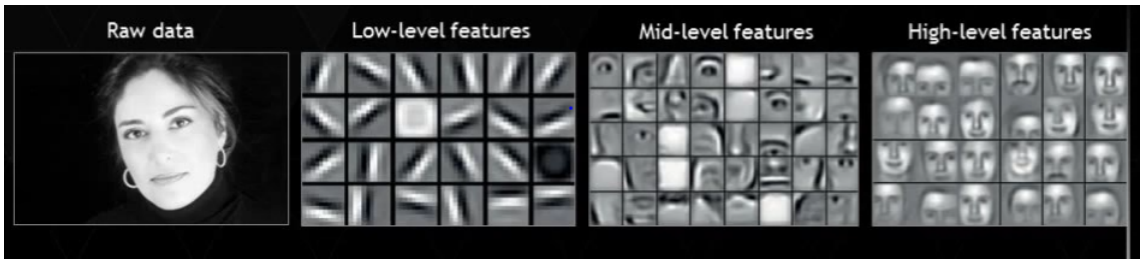

Figure 2: Some examples of what low, mid and high level features can look like. A Deep Learning network has the ability to extract features of all levels while engineered features mostly are low level features. Image from [27].

2.1.2 Supervised and Unsupervised Learning

Supervised learning is when the input, X, and output, y, is given and a mapping function, f, is to be learnt such that $y = f(X)$. The mapping function can, for example, be approximated by a Neural Network and it is found through optimisation based on the comparison of the known output and the predicted output. If the difference between known and predicted output, as defined by the chosen loss function (see Section 2.2.5.1), reaches an acceptable level, a satisfactory mapping function has been found.

Unsupervised learning has no output to help train a mapping function and cannot directly be used for a classification problem. Instead it can be used to find underlying structure and patterns of the data. It can solve a simple clustering problem to group data by its similarity or find anomalies in the dataset.

Unless otherwise stated, all learning tasks considered in this thesis are supervised learning tasks. The input consists of bookshelf images or images of cropped books and the output is either masks of where books are located or their titles and author name. The images are annotated and labelled so that the output can be compared with its ground truth. More on this in Section 3.

2.1.3 Underfitting and Overfitting

Under- and overfitting are measures of how well a model can generalise to the data. If a model is underfitting it lacks the capacity to capture the underlying structure of the dataset. It would not

perform well on either the training dataset or on new data. A model that is overfitted however has more capacity than can be justified, and would perform very well on a training dataset but not necessarily on new data. In Figure 3 the difference between under and overfitting is visualised.

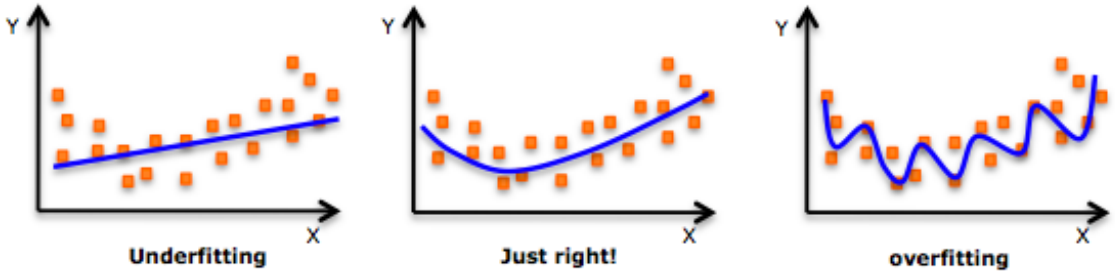

Figure 3: A visualisation of the main difference between underfitting and overfitting where the dots are data points and the blue line corresponds to the model's predictions. Image from [19].

Figure 3 visualises what it might look like when the model is tested after the training process, but it is possible to see when a network is starting to overtrain during the actual training too. If the dataset is divided into training and validation data, the network is trained on the training data and tested on the validation data. A loss function (see Section 2.2.5.1) is calculated after each epoch and tells us how well the model fits the data. As long as the loss function decreases for both the training data and the validation data, it is a good sign. Often the training loss keeps getting better while the validation loss eventually stops to improve (and often gets worse). This is an indication that the network is starting to overtrain.

To prevent a model from overfitting one can use *early stopping* which means that the training gets interrupted once the performance on the validation data stops to improve. Another approach is to regularise the training process, for instance by using dropout layers that nullifies the output of the neurons with some probability so that the network is forced to learn more generalised patterns, rather than memorising it. Other options include: changing the network structure to reduce the number of parameters, using parameter regularisation, adding noise to input, weights or output, or to use bigger datasets with more generalised data. If a bigger dataset is not available, then more data can sometimes be synthesised to a degree through data augmentation [8].

2.1.4 Evaluation

There are many ways to evaluate a model, a few common ones for classification problems are calculating accuracy, recall and precision when comparing predicted labels with the ground truth. The object detection is evaluated with mean average precision (mAP) where a measure of Intersection over Union (IoU) is necessary.

2.1.4.1 Accuracy, Recall, Precision

The accuracy ratio is given by the number of correct predictions over the number of samples. In addition, to give us an idea of what is hard to predict one can calculate the precision and recall of the predictions. They are defined by:

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}, \quad \text{Precision} = \frac{tp}{tp + fp}, \quad \text{Recall} = \frac{tp}{tp + fn} \tag{1}$$

where tp is the number of true positives, tn is the number of true negatives, fp is the number of false positives and fn is the number of false negatives. The precision implicates how well it finds

correct positives while the recall implicates how well it finds **positives**.

For an optimal model the accuracy, precision and recall should be as close to 1 as possible.

2.1.4.2 Mean Average Precision, mAP

When working with object detection a common approach to evaluate the detections is to calculate the mean Average Precision (mAP). The Average Precision (AP) of a certain class is calculated by measuring the precision for all occurrences of the object class in the test dataset. Accordingly, the mAP is the mean AP for all object classes.

2.1.4.3 Intersection over Union, IoU

Intersection over Union is, much like its name implies, a ratio that is calculated by the intersection of two detections over the union of the two detections. A detection in this case is a bounding box around a detected object in an image. With a measure of IoU, one can determine whether or not two bounding boxes in the same image actually show the same object, or one can compare a bounding box with ground truth annotations to calculate accuracy or precision. If the IoU is bigger than some threshold, the two bounding boxes will be interpreted as containing the same object. A commonly used threshold is 0.5 which then means that if the intersected area is at least half as big as the combined area, these detections show the same object. Some recent papers have benchmarked their models with multiple thresholds on the COCO dataset. They calculate the mAP with an IoU of 0.5, the mAP with an IoU of 0.75 and one as an average mAP from the 10 different mAP:s using IoU:s from 0.5 to 0.95 with a step size of 0.05. A high value of the average mAP indicates that the localisation works well [4].

2.2 Convolutional Neural Network, CNN

A Convolutional Neural Network consists of an input layer, a number of hidden layers, and an output layer. Because of its spatial dependence it has several applications within image analysis and classification problems. The input layer is usually images one wants to classify or somehow extract information from. The hidden layers process the input image to extract useful information to finally create an output which is used in different ways. For example, in classification problems the output indicates what class the input image most likely belongs to. At least one of the network's hidden layers must be a Convolutional layer to be called a Convolutional Neural Network [9]. An example of such a network is shown in Figure 4.

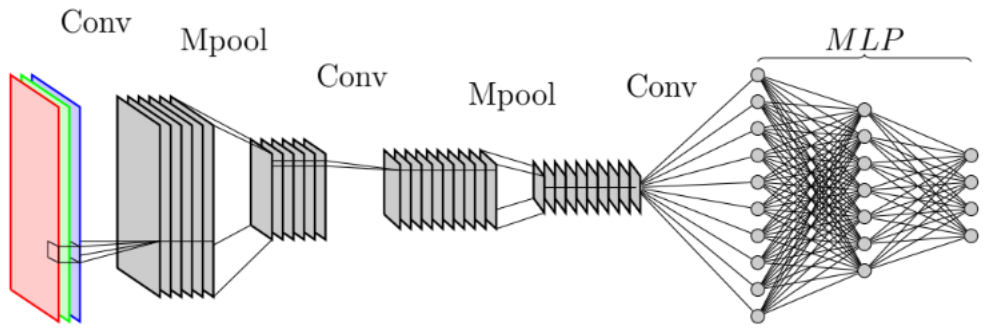

Figure 4: The structure of a Convolutional Neural Network. It takes an image as input and processes it through a number of hidden layers consisting of convolutional layers, max pooling layers and a multi layer perceptron i.e. a series of fully connected layers which outputs a predicted class. Image from [9].

Typical hidden layers are convolutional layers, pooling layers and fully connected layers mixed with activation functions. In the end, the network is evaluated with a loss function and updated with an optimiser. This is where the learning and updating of parameters takes place and it is described in Section 2.2.5.

2.2.1 Convolutional Layer

A Convolutional Layer has filters that are convolved over the image. The image pixels are multiplied with the filter values and the sum creates a new output image that is the input image for the next hidden layer. Figure 5 visualises how this works.

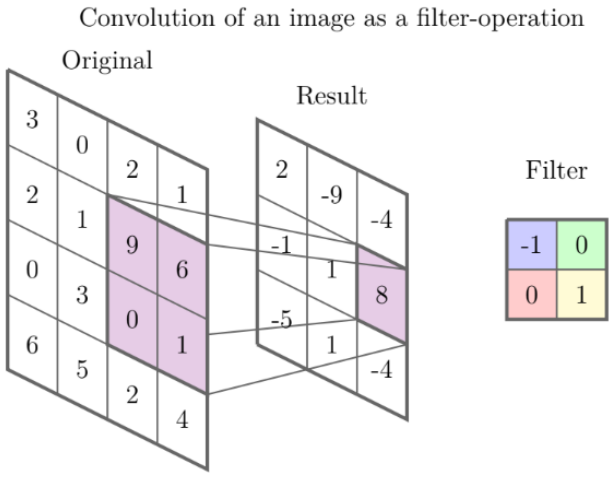

Figure 5: The principle of a Convolutional Layer. The filter is convolved over each part of the image, creating a new image. N.B. as the convolution filter is folded, think of the inverse filter in this case, as if the filter was a 2x2 matrix, to calculate the result ($\text{inv}([-1{,}0;0{,}1]) = [1{,}0;0,-1]$). Image from [9].

When designing a Convolutional layer there are multiple hyper-parameters to set e.g. the number of filters, filter size, padding and stride. In Figure 5 a filter of size 2x2 is visualised. The resulting image is slightly smaller than the original. Padding can be used if one wants to prevent this. With padding the original image is expanded with zeros so that the resulting image retains its size. The stride parameter is set based on how big steps the filter should take. A stride of 1x1 has been used in Figure 5, i.e. the filter is moved one step at the time. Bigger strides let the filter skip some of the positions, which also results in a smaller output image.

2.2.2 Pooling Layer

There are different pooling layers such as max pooling layers and average pooling layers. Pooling Layers are similar to Convolutional Layers but instead of multiplying values and summing them, only some metric such as the maximum or average value within the filter is extracted. Figure 6 shows the principle of pooling using a maximum metric.

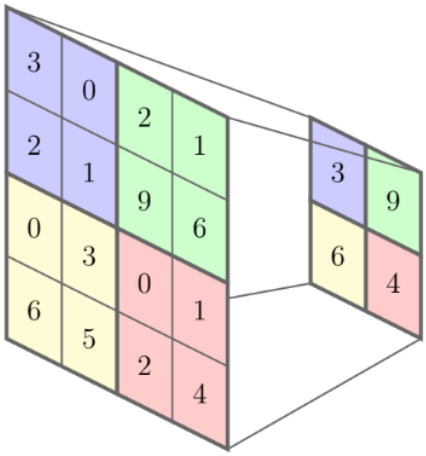

Figure 6: A figure of how a max pooling layer works. With filter size 2x2, the biggest element in each 2x2 square is extracted to the next layer. Image from [9].

Similar to the Convolutional layer a pooling layer have the hyper-parameters filter size, padding and stride. There is only one filter as opposed to Convolutional layers that can have many. The filter size determines how big area each value will be extracted from, for instance in Figure 6 a 2x2 filter size is used. Padding can be used here as well but with big negative numbers instead of zeros so that they do not get chosen by the max pooling if there are negative numbers present. Each step is determined by the stride parameter, just like in the Convolutional layer. In this example there is a 2x2 stride.

2.2.3 Fully Connected Layer

In a Fully Connected Layer, every node in a layer is connected to each node in the next. Propagation through these layers consists of a matrix multiplication of the output from the previous layer with the layer weights. An activation function (see Section 2.2.4) is then applied on the result to produce the input to the next layer or the output of the neural network. The last Fully Connected Layer outputs a predicted probability for each label. Figure 7 illustrates how this works.

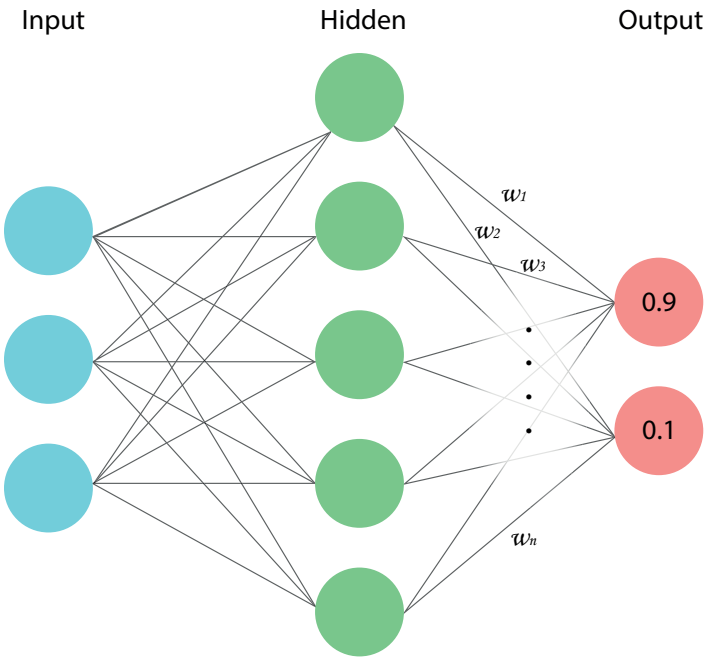

Figure 7: The structure of a fully connected layer, each node is connected with all the nodes in the next layer. Weights are added to each link and the probability of belonging to each class is given.

2.2.4 Activation Functions

Activation functions are used in neural networks to determine the output of each node. Activations with nonlinear properties can create a model that is able to find nonlinear and complex mappings between input and output which makes the Neural Network more powerful. Figure 8 shows a few examples of activation functions.

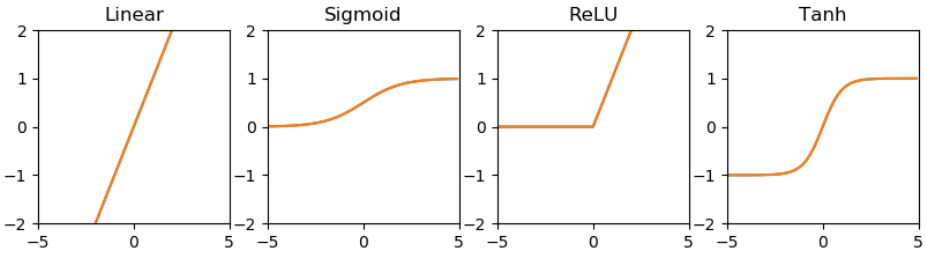

Figure 8: Four examples of activation functions.

These are commonly used and there are many other activation functions to choose from. In this thesis, only sigmoid and the Rectified Linear Unit (ReLU) have been used.

Sigmoid

The sigmoid activation function is defined by $g(x) = \frac{1}{1+e^{-x}}$ and is especially useful when one wants to interpret the output as a probability because when applied, the output ranges from 0 to 1. The graph of the function can be found in the mid left image in Figure 8.

Rectified Linear Unit

The rectified linear unit activation function is a nonlinear function, though it is piece wise linear and therefore still has many properties that makes it easy to optimise. Applied to a linear transformation the output will be nonlinear. The function is defined by $g(x) = \max(0,x)$ and the graph can be found in the mid right graph in Figure 8.

2.2.5 Training a CNN

2.2.5.1 Loss Function

The loss function in a Neural Network is used to optimise the parameters. It is essentially what a machine learns from. A Neural Network strives to minimise the loss function as the loss is a scalar that indicates how well the parameters of the Neural Network accomplish the task they were given to do. The loss function decreases as more correct predictions are made by the Neural Network. The choice of loss function is important because of its significance during training but there are numerous loss functions to choose from.

To begin with there are different categories of loss functions, such as regression, binary classification and multi-class classification loss functions. The regression loss functions use the error between the input and prediction to calculate, for instance, the Mean Squared Error. Binary classification loss functions are suitable for problems where the data belongs to two classes. An examples of such loss functions is Binary Cross-Entropy. Among multi-class classification loss functions there are for instance Categorical Cross-Entropy and Triplet loss. These are adequate loss functions when there are two or more possible classes.

As there are a large number of loss functions, each and every one of them will not be explained here, only the ones used for this thesis.

Cross-Entropy Loss

Two versions of cross-entropy loss are commonly used, namely binary and categorical cross-entropy loss. Binary is used for "yes or no" problems, or in other words, when there are two possible classes, 0 and 1. Each sample gets a predicted probability from a sigmoid activation function between 0 and 1 of belonging to class 1. The binary cross-entropy loss function measures how far the probability is from the true class. It is calculated through

$$L(y, \hat{y}) = -\frac{1}{N}\sum_{i=0}^{N}(y_i \log(\hat{y_i})) + (1 - y_i)\log(1 - \hat{y_i})), \tag{2}$$

where y is the true class, $\hat{y}$ is the predicted probability of belonging to class 1 and N is the number of samples [13]. For a sample where $y = 0$, the contribution is $-\log(1 - \hat{y})$ which is close to zero if the prediction is close to zero, but very big if the prediction is close to one.

The categorical cross-entropy is a loss function that is used for classification among many possible labels, not only two. Instead of a sigmoid activation function, a softmax function is used. The softmax function acts like a sigmoid for many classes. This loss function can be calculated with

$$L(y, \hat{y}) = -\sum_{j=0}^{M}\sum_{i=0}^{N} y_{ij} \log(\hat{y}_{ij}), \tag{3}$$

similarly to equation (2), but over all the M possible classes [13]. The true class is still $y = 1$ and all other classes are $y = 0$, meaning that the true class is represented as a one-hot encoded vector.

Mask R-CNN Loss

Mask R-CNN is used for object detection and it uses a multi-task loss L, a combination of three loss functions as described below.

$$L = L_{cls} + L_{box} + L_{mask} \tag{4}$$

L_{cls} is the loss function that optimises the classification of each Region of Interest (ROI), it is

defined by $L_{cls}(p, u) = -\log p_u$ where p is a discrete probability distribution $p = (p_0, ..., p_K)$ over $K + 1$ categories and u is the true class.

L_{box} optimises the bounding box and is given by $L_{box} = \lambda[u \geq 1]L_{loc}(t^u, v)$ where λ is a hyper-parameter that controls the balance between the losses, normally set to $\lambda = 1$, $[u \geq 1]$ is the Iverson bracket indicator function which is 1 when $u \geq 1$ and 0 otherwise. t^u and v are the predicted bounding box and the anchor bounding box respectively. L_{loc} is defined by

$$L_{loc}(t^u,v) = \sum_{i \in x,y,w,h} \text{smooth}_{L_1}(t_i^u - v_i), \text{ where } \text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \quad (5)$$

The last loss function, L_{mask}, is calculated after a per-pixel sigmoid is applied as an average binary cross-entropy loss for each class [7], [10].

Triplet Loss

Triplet loss is a fairly new approach that has been used mainly for face recognition with good results. It trains a network that learns to create embeddings, i.e. a vector of values, for each sample. A sample is chosen together with a positive (a sample of the same class) and a negative (a sample of another class), and the distance between the embeddings is to be minimised for the anchor and the positive and maximised for the anchor and the negative. Figure 9 visualises what is being learnt.

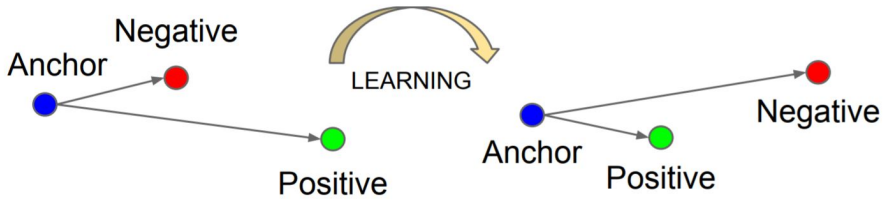

Figure 9: An illustration of what triplet loss is training the network to do. Image from [26].

The loss function in this case is

$$L = \sum_{i=1}^{N} \left[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha \right]_+ \quad (6)$$

where f is an embedding for a sample x. The indices a, p and n indicates if the sample is an anchor, a positive or a negative. α is a minimum margin of separation between positive and negative pairs and N is the cardinality of possible triplets [26].

2.2.5.2 Optimiser

The loss function gives the network an indication of how good the predictions are, but it does not in itself improve the network, that is the optimisers' task. There are many different optimisers and the ones used in this thesis are Stochastic Gradient Descent (SGD) and Adam.

Stochastic Gradient Descent

SGD is used for the object detection part. It is an optimiser that calculates a gradient based on the loss of a single random sample, thus stochastic sample. SGD is often used interchangeably with mini-batch SGD, meaning that the stochastic sample is a mini-batch and the parameters are updated based on the mean contribution from all samples within a mini-batch. In this report, SGD and mini-batch SGD mean the same thing as well. SGD update is defined by

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \mathbf{g} \tag{7}$$

where $\mathbf{w}$ are the weights, α is the learning rate and $\mathbf{g}$ is the computed gradient estimate given by

$$\mathbf{g} \leftarrow \frac{1}{m} \nabla_{\mathbf{w}} \sum_{i=1}^{m} L(\mathbf{x}^{(i)}, \mathbf{w}, \mathbf{y}^{(i)}) \tag{8}$$

which is calculated over a mini-batch with size m, samples $\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(m)}\}$ and corresponding targets $\mathbf{y}^{(i)}$ [8].

Adam

Adam, a name derived from *adaptive moment estimation*, is based on AdaGrad, which individually adapts the learning rate for each model parameter, and another optimiser called RMSProp, an algorithm that prevents too rapid learning rate decays which might lead to finding local minima. Adam is a an algorithm that is fairly robust and well suited for problems with much data and parameters. It is based on adaptive estimates of lower-order moments and the parameter update is calculated as

$$\begin{aligned} t &\leftarrow t + 1 \\ \mathbf{s} &\leftarrow \rho_1 \mathbf{s} + (1 - \rho_1)\mathbf{g} \\ \mathbf{r} &\leftarrow \rho_2 \mathbf{r} + (1 - \rho_2)\mathbf{g} \cdot \mathbf{g} \\ \hat{\mathbf{s}} &\leftarrow \frac{\mathbf{s}}{1 - \rho_1^t} \\ \hat{\mathbf{r}} &\leftarrow \frac{\mathbf{r}}{1 - \rho_2^t} \\ \nabla \mathbf{w} &= -\alpha \frac{\hat{\mathbf{s}}}{\sqrt{\hat{\mathbf{r}}} + \delta} \\ \mathbf{w} &\leftarrow \mathbf{w} - \nabla \mathbf{w} \end{aligned} \tag{9}$$

with each time step t, biased moment variables $\mathbf{s}$ and $\mathbf{r}$, corrected moment variables $\hat{\mathbf{s}}$ and $\hat{\mathbf{r}}$ and exponential decay rates ρ_1 and ρ_2 for moment estimates. t, $\mathbf{s}$ and $\mathbf{r}$ are initialised with $t = 0, \mathbf{s} = 0, \mathbf{r} = 0$ [14],[8].

2.2.5.3 Transfer Learning

As Deep Learning has grown popular there are lots of huge datasets available online and some users have trained neural networks on these datasets for multiple hours, days or even weeks. These weights are often good feature detectors, even though they might be trained on another kind of

dataset, because they have probably learnt low level features such as edges and corners that are useful for many different datasets. This is something that can be utilised to save many hours of training and to improve a network's performance. It might also decrease the amount of data needed to train a network from scratch.

One can use the weights as start weights, just as they are. Then the network will still update them during training. Another way to use them is locking the weights of the first layers, only to fine-tune the last few on a new dataset. This way the low level features will be preserved and the network will only need to update some of the weights.

2.3 Object Detection Networks

CNN:s are powerful tools for classifying images and have been used for a few decades. One significant limitation for CNN based classifiers is when the images contain several objects. A CNN cannot localise each object in an image and classify them. To solve this, one can use the Sliding window algorithm where a window is slid over an image while each segment is classified. This results in a massive number of different regions for each image. Not only would it take a lot of time to classify each region but one would need to do this with multiple scales of the window to correctly detect objects which makes it unreasonably heavy to compute. Over the last couple of years several kinds of networks have been researched, such as R-CNN, SPPnet, Fast R-CNN and Faster R-CNN, which all are much faster than the Sliding window algorithm.

2.3.1 Region-based Convolutional Neural Network, R-CNN

Ross Girshick et al. used another approach and developed an object detection system which they called R-CNN [6], short for Region-based Convolutional Neural Network. They used Selective Search to extract a large number of Regions of Interest (ROI). Selective Search is a region proposal algorithm which groups similar pixels to segments. The similarity measures are based on colour, texture, size and shape compatibility [29]. The regions are classified with a CNN. Though this technique made it possible to detect objects in images without a countless number of regions, as in the Sliding window approach, processing all the ROI still takes a lot of time and the network does not learn better region proposals as Selective Search is fixed.

2.3.2 Spatial Pyramid Pooling Network, SPPnet

Another model was researched to prevent having to go through all the Selective Seach ROI for each image, namely Spatial pyramid pooling networks (SPPnets). Instead of using Selective Search they used a CNN on the whole image to extract a feature map from which ROI could be selected. This model is 24-102 times faster than R-CNN. Because the training of SPPnets is a multi-stage pipeline this limits the model's accuracy. The CNN that extracts feature maps is frozen during training of the fine-tuning algorithm [11].

2.3.3 Fast Region Convolutional Neural Network, Fast R-CNN

Unlike SPPnets, Fast Region Convolutional Neural Networks use single-stage training which means that every layer can be trained simultaneously using a multi-task loss function. Fast R-CNNs do not have the Spatial pyramid pooling layer but a single ROI pooling layer instead, which essentially is a max pooling layer applied on the ROI. Fast R-CNNs turned out to be much faster to train than SPPnets and also more accurate [7].

2.3.4 Faster Region Convolutional Neural Network, Faster R-CNN

The region proposal method that selects ROI from the feature map in SPPnets and Fast R-CNNs was found to be very time consuming. Ross Girshick et al. had an idea on how to speed up the detection, which resulted in what they call a Faster Region Convolutional Neural Network. Instead of the region proposal method in SPPnets and Fast R-CNNs, Faster R-CNNs have a Region Proposal Network (RPN) that is trained to find good ROI from the feature map. After ROI have been found, the rest of the network is basically a Fast R-CNN which uses the same feature map for the classification as the RPN used to find ROI [23].

2.3.5 State of the Art Models

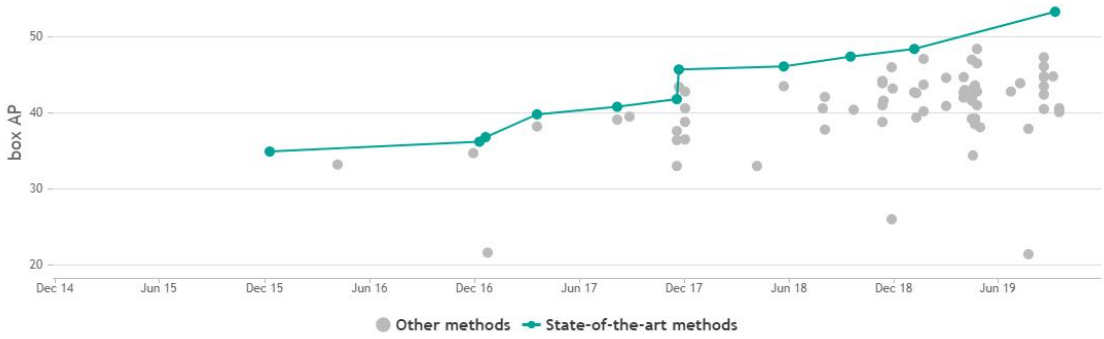

Figure 10: This is a graph that visualises the results from different Object Detection Networks tested on a COCO test-dev dataset. The one with best box AP result so far is a Cascade Mask R-CNN. box AP is a measure of how well a model performs on the COCO test-dev dataset and is calculated on COCO:s evaluation server. Image from [5] and more information on the Cascade Mask R-CNN model can be found in [17].

As object detection grows more popular, more developers try new networks from which better and better results are achieved. The graph in Figure 10 shows some results from different kinds of networks that are all tested on the same test dataset to give a fair comparison. Among these networks one for instance finds Mask R-CNN, YOLO, SSD, RetinaNet and many different versions of these using various backbones, i.e. feature extractors, such as ResNet, VGG, Cascade and HRNet just to mention a few.

2.3.5.1 Mask Region Convolutional Neural Network, Mask R-CNN

The previously described object detection networks all use bounding boxes around the detected object which can be problematic for objects that get overlapping bounding boxes to a large extent. Mask R-CNN was created, with the idea of adding a branch to Faster R-CNN that predicts an object mask. This branch is added to the architecture in parallel to the classification and box offset prediction. It is applied on each ROI and consists of a Fully Connected Network (FCN) that predicts a pixel-by-pixel segmentation mask within the ROI. The mask is binary and represents whether or not the pixels belongs to a detected object [10].

2.3.5.2 You Only Look Once, YOLO

You Only Look Once is a one stage method that does detection and classification in the same step. The previously mentioned methods are two stage methods that first detects objects and then classifies them. The YOLO architecture consists of a feature extractor followed by fully connected

layers. This makes YOLO much faster than the other methods but as a result it does not have as good accuracy. It can have trouble with small objects because of its implementation [21],[22]. This is probably a suitable method when speed is an important aspect.

2.3.5.3 Single Shot Detector, SSD

The Single Shot Detector is also a one stage method that first uses a feature extractor, just like the other models, but then adds Convolutional layers of multiple scales. This helps it find objects of various scales. This method is also very quick but not as fast as YOLO [16].

2.3.5.4 RetinaNet

Images often contain lots of background and only a fraction of the image contains the actual objects to be detected. This makes the one stage methods inefficient as a lot of the computations are made on the easily classified background. An approach to solve this is using the RetinaNet model which uses Focal Loss that handles the problem with class imbalance. Focal Loss down-weights easy examples so that it can focus on the hard examples during training. RetinaNet outperformed many object detection models when the article presenting RetinaNet was published in 2017 [15].

3 Methodology

"If we could build computers that could understand what's in an image and could tell a blind person who otherwise couldn't see that image, that would be pretty amazing as well" - Mark Zuckerberg

3.1 Data gathering

Images of book spines were mainly photographed at three libraries in Lund, *Lunds Stadsbibliotek*, *Library of Mathematics* and *Universitetsbiblioteket - Lund University*. Some were also photographed at the author's home. 36 images were labelled with titles and authors, which together contained 276 different books. An additional 10 images were labelled to be used as test images, they all contained books that the network had been trained on, but from a new angle or with a new distribution. A list of all the 276 labelled books can be found in the appendix.

To label the images, Wada's implementation of *labelme* [31] was used. It is a graphical image polygonal annotation tool developed in 2016. Each book spine was marked with a polygon whose coordinates were saved together with the book title and author in json file format.

Furthermore, 24 images were labelled only with a segmentation mask, 14 of them to improve the object detection network and 10 to use as test images, to see how well the network performs on books that it has not seen at all. These were images of new books that were not used to train the classification network as they were not labelled with title and author, only with a mask.

In each image there were 5-65 books and 18 books on average. Examples of such annotated images can be found in Figure 11.

Figure 11: Two examples of annotated images. To the left are images of books and to the right are the annotated books with a polygon. Each colour is associated with a unique book in that image.

3.1.1 Data augmentation

Neural Networks generally give better results with more and diverse data because then it tends to learn generalised characteristics of its training data instead of memorising each detail, that way it becomes more robust. Therefore one would like to have thousands of different images but

the labelling procedure is in general the most time consuming part of the data gathering process. This can be solved to some degree through data augmentation. It is a relatively easy and time saving method to expand the dataset. One way to do this is creating distorted copies of the existing labelled data. Such distortions can for example be horizontal and vertical flips, rotations, brightness and scale changes, shears, translations and added noise.

Another method of augmenting images is cropping books and pasting them into different background images. The object detection algorithm then learns to find books in various environments and to recognise more negative samples, thus learns from more examples that are not book spines. An example of such a negative sample is the mobile phone or the computer in the upper left image in Figure 12. It would not be very surprising if the network wanted to classify them as books as they have fairly similar shapes.

This kind of data augmentation was done which created 2280 different images from the original 36+14 annotated images. All the images were resized uniformly so that the longest side of each image was 1024 pixels long. These were used to train the Mask R-CNN. Four examples with four different augmentation variants are presented in Figure 12.

Figure 12: Examples of augmented images. In the upper left image, books have been pasted onto a background image, trying to mimic a bookshelf. The upper right image has noise and brightness changes added to it. In the lower images, one has been flipped upside down and one has gotten cropped books pasted onto it randomly.

To train a classifier, crops were generated from an augmented dataset and rotated so that every image contained one book, horizontally aligned in the middle of the image. The rotation was made based on the dominant direction of the image, utilising the typical shape of a book that is more elongated than it is wide. The dominant direction was calculated through Principal Component Analysis (PCA) which finds the direction with biggest variation. As it is a quick fix that can be done on both training data and test data, this was performed with hope that the classifier would learn the books easier, as it would not have to be rotation invariant. Though it still have to learn two distributions of each book; one normal and one upside-down, as the PCA cannot know which is which and both distributions occur.

During the training of the Mask R-CNN several checkpoint models were saved which each could

be used to crop more examples of each book. As the crops from a Mask R-CNN might loose some information, especially in the corners of the books, these crops could be used for the training process as they generalise the dataset. Crops were generated both from the object detection network's output and directly from image annotations. Each crop was resized to the size of 200x200 pixels. Examples of such images can be found in Figure 13.

Figure 13: Two different crops of the same book. The left one has been cropped with the Mask R-CNN and some information has been lost around the corners. The right one has been cropped directly from the annotated segmentation mask. Both kinds of crops were used to train the classifiers.

3.2 Training Neural Networks

3.2.1 Mask R-CNN

The Mask R-CNN was trained with Abdulla's implementation of Mask R-CNN with transfer learning. There are weights pretrained on the COCO dataset available on his GitHub account [1] which were used for this model. The network uses ResNet-101 as backbone which is a 101 layers deep CNN. It has been trained on 1.28 million images and can classify images into 1000 categories [12]. Abdulla has added a few layers to make it work like a Mask R-CNN and trained the network on 81 different classes (80 + background). The last four layers are excluded as they require 81 classes, and in this case there are only two; background and books.

The network was trained on a dataset with 2035 augmented training images and 245 validation images and it uses SGD as optimiser. The learning rate was set to 0.001, as Abdulla suggests, and a small batch size of two images was used as each image takes about 6 GB RAM to process. An IoU threshold of 0.5 was used during training and a >90% detection confidence was required. The network was trained for 214 epochs.

3.2.2 CNN with Categorical Cross-entropy Loss

The Neural Network for recognition of books was trained in two different ways. One CNN that outputs a class based on the categorical cross-entropy loss function and the other was a CNN that optimises embeddings that are used as feature vectors for each image. The first mentioned was implemented with Keras using a three Convolutional layers, two maxpooling layers and two fully connected (dense in Keras) layers. More details on the network architecture can be found in Table 1.

When choosing parameters and doing hyper-parameter tuning, several things were tried. For instance, adding dropout layers and adjusting the number of Convolutional layers. Padding such

that the layer size stayed the same through a Convolutional layer was tested and class weights were used to counteract the class imbalance. Parameters like number of filters and filter size in a Convolutional layer, number of units in the dense layer and batch size were varied. The version with best result on the test dataset however is the one presented in this report.

The CNN was optimised with Adam and used a batch size of 32. In total there were 12573 training images, 8-96 images of each book. 20% of the images were used for validation. It was trained until interrupted by the early stopping function which was activated when the validation loss function no longer decreased. No padding, dropout or class weights were used. Every layer was used ReLU as activation function, except the last one that had a softmax activation function. No transfer learning was performed for this network.

Table 1: The architecture of the classification network with categorical cross-entropy that takes a crop image as input and outputs a class.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 200, 200, 3)	0
conv2d_1 (Conv2D)	(None, 198, 198, 16)	448
max_pooling2d_1 (MaxPooling2D)	(None, 99, 99, 16)	0
conv2d_2 (Conv2D)	(None, 97, 97, 16)	2320
conv2d_3 (Conv2D)	(None, 95, 95, 8)	1160
max_pooling2d_2 (MaxPooling2D)	(None, 48, 48, 8)	0
flatten_1 (Flatten)	(None, 18432)	0
dense_1 (Dense)	(None, 500)	9216500
dense_2 (Dense)	(None, 276)	138276
Total params:	9,358,704	
Trainable params:	9,358,704	
Non-trainable params:	0	

3.2.3 CNN with Triplet Loss

The network that optimises embeddings was implemented using Sandberg's TensorFlow implementation of FaceNet, an embedding based Recognition and Clustering method that has received good results on images of faces. Inception ResNetv1 was used as backbone whose specifics can be found in [28]. As it has a complicated architecture with many blocks of layers, it will not be further explained here. Pretrained weights are available online and Sandberg has modified two weight sets to fit his implementation, namely the ones trained on the datasets CASIA-WebFace and VGGFace2.

The parameters of the Triplet loss network were not varied as much as the ones in the categorical cross-entropy loss network as it takes longer time to train. The layers was not changed much as it already has a complex backbone, thereby other parameters were mainly focused on. Learning rate, batch size, and dropout probability were altered. The optimisers Adam and AdaGrad were tried. Also, different embedding sizes and the two pretrained weight sets were experimented with.

The best received results were given from a model using VGGFace2, trained for 4638 epochs. The learning rate was initially set to 0.01, the batch size was 90, the embedding size was 512 and the optimiser Adam was used in this case with no dropout. The triplet loss margin was 0.2.

3.3 Evaluation and Benchmarking

To evaluate the object detection the mAP of a test dataset was calculated. Two different test datasets were used, one with books that the network had never seen and one with books that it had seen. Each dataset consisted of 10 images. The first named test dataset had more thin books and in general more books in each image compared to the second test dataset, the first can therefore intuitively be interpreted as "difficult images" and the second as "easy images". The mAP of the datasets was calculated with the weights giving the lowest validation loss. The IoU threshold was varied like researchers have done in recent papers, as explained in Section 2.1.4.3. The mAP with different image sizes was also calculated.

The crops from the easier dataset were used to benchmark the classification networks. As the CNN with categorical cross-entropy outputs a predicted class, an accuracy score could directly be calculated. Recall and precision could be calculated as well but as each book in the test dataset generally occurs only once, the results would not provide much information.

The network with triplet loss was evaluated by comparing the embeddings. The embedding of a cropped book was compared with other books that the network had been trained on. They were both compared directly to other book embeddings and to mean embeddings within each class. Each sample was classified as the book with the embedding that has the smallest distance to the sample embedding. The distance was calculated with cosine similarity, the L2 norm and the L1 norm, and the accuracy of the predictions was calculated.

The accuracy of both recognition models was modified so that false positives from the Mask R-CNN would not interfere with the results. The raw accuracy gives an indication of how well the whole object detection and recognition model works on the test set and the modified gives an indication on how good the recognition is in itself. The results are also compared to annotation crops, i.e. "perfect crops", to compare whether the model performs much worse on imperfect crops from the Mask R-CNN or not.

4 Results

"Unfortunately, we have biases that live in our data, and if we don't acknowledge that and if we don't take specific actions to address it then we're just going to continue to perpetuate them or even make them worse." - Kathy Baxter

4.1 Object Detection with Mask R-CNN

After training a network with Abdulla's implementation of Mask R-CNN for 214 epochs the lowest validation loss was found at epoch 126. The test results on the two test datasets with these weights can be found in Table 2 and 3. In the left column of the Tables, one finds the size of the biggest image side, the images are resized uniformly and might have been cropped in the pre-processing stage, therefore the shorter image side can differ. The very best results were 0.860 on the difficult test data and 0.944 on the easier one. Figure 14 visualises the loss functions on the training and validation datasets after each epoch and there is a marker where the lowest validation loss was received. The number of books detected in the easier test dataset and the number of false positives detected can be found in Table 4.

In the rest of the results section, only the easier test dataset will be used as this is a dataset where the books have been labelled as individual books with title and author and these are books that the CNN:s have been trained on. The difficult test dataset consists of books that the Mask R-CNN had not seen and were only labelled as "book" and can not be used to test the recognition.

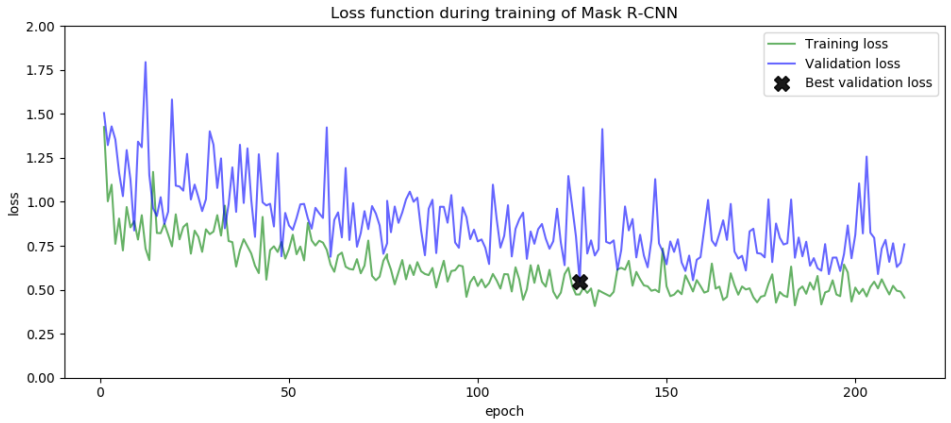

Figure 14: The loss function during training of Mask R-CNN for 214 epochs. The cross shows where the lowest validation loss was found, namely at epoch 126.

Table 2: These are the mAP results given on the more difficult test dataset. Here, mAP is calculated as the mean mAP when calculating mAP for 10 different IoU:s. The IoU ranges from 0.5 to 0.95 with a step size of 0.05. mAP50 and mAP75 are the single mAP:s with IoU = 0.5 and IoU = 0.75 respectively. The best mAP:s have been marked with bold text.

Longest image side	mAP	mAP50	mAP75
4608	0.600	0.854	**0.738**
1024	**0.601**	**0.860**	0.730
512	0.562	0.839	0.664
256	0.421	0.708	0.483
128	0.08	0.250	0.037

Table 3: These are the mAP results given on the easier test dataset. mAP is calculated as the mean mAP over several IoU:s. mAP50 and mAP75 are the mAP:s with IoU = 0.5 and IoU = 0.75 respectively. The best mAP:s have been marked with bold text.

Longest image side	mAP	mAP50	mAP75
4608	0.689	0.942	**0.859**
1024	**0.693**	0.942	0.852
512	0.674	**0.944**	0.829
256	0.587	0.857	0.725
128	0.281	0.501	0.304

Table 4: For the easier test dataset with longest image side defined by the left column, the Mask R-CNN found a number of books among which a few were false positives.

Easy test data	Number of books	False Positives
Mask R-CNN crops		
4608	116	2
1024	116	3
512	114	1
256	111	2
128	71	4
Crops from annotation	117	0

4.2 Classification with Categorical Cross-entropy Loss

After Mask R-CNN had found books in the easier test dataset, where every book is a known book, the crops generated from the segmentation mask were classified in a CNN with categorical cross-entropy loss. The results are shown in Figure 15 and in Table 5. The crops were generated from different image sizes. As the Mask R-CNN is not perfect, it did not find all the books. The quantity of books that were found is shown in the Number of books column. It also generated a few false positives, fp. The false positives were either crops of something in the background or a part of a book that had IoU<0.5 with the annotated mask. They are automatically misclassified. The accuracy score includes the fp, giving a measure of how good the object detection and the recognition method are together and the Modified accuracy excludes the fp, giving an indication of how good the CNN is in itself. The table also have the scores of crops directly from the annotation, which do not contain any fp.

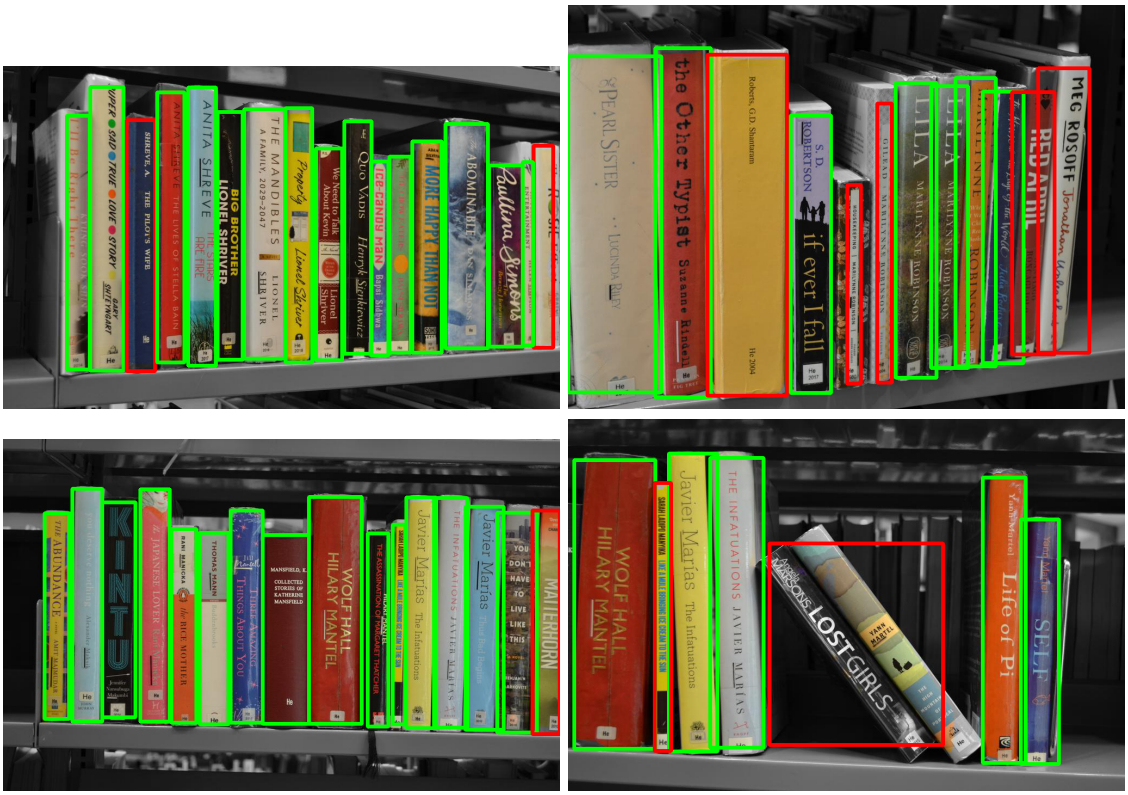

Figure 15: Images from the easier test data set where each detected sample has been marked with a bounding box, including false positives. Correct classifications are coloured green and misclassifications are coloured red.

Table 5: The results of the CNN with categorical cross-entropy loss from Mask R-CNN crops and annotation crops. The columns show the number of books that were found, including false positives, fp, accuracy of the prediction including fp and a modified accuracy excluding fp.

	Accuracy	Modified accuracy
Mask R-CNN crops		
4608	0.836	0.851
1024	**0.879**	**0.903**
512	0.825	0.832
256	0.694	0.706
128	0.310	0.328
Crops from annotation	**0.949**	**0.949**

4.3 Classification with Triplet Loss

The best Triplet loss CNN was trained for 4638 epochs on the same training dataset that were used for categorical cross-entropy loss. The VGGFace2 weights were used as pretrained weights in this case. The network was used to create embeddings for the test dataset which was compared to the embeddings for training samples. They were compared to mean embeddings that were calculated from the training dataset. They were also compared to 10 samples of each book, if there were 10 samples, and classified as the book whose embedding it was closest to. The distance between the embeddings were calculated through Cosine similarity (CS), L1 norm and L2 norm. The results on the comparison of mean embeddings can be found in Table 6 and the results on the comparison

of 10 samples can be found in Table 7.

Table 6: The results of the Triplet loss network when comparing each test sample with mean embeddings within each class. Both raw accuracy and modified accuracy that ignores false positives from the Mask R-CNN are included.

	Accuracy			Modified accuracy		
Mask R-CNN crops	CS	L1 norm	L2 norm	CS	L1 norm	L2 norm
4608	**0.888**	**0.888**	**0.888**	**0.904**	**0.904**	**0.904**
1024	0.879	0.879	0.879	0.903	0.903	0.903
512	0.877	0.868	0.877	0.885	0.876	0.885
256	0.883	0.874	0.883	0.899	0.890	0.899
128	0.366	0.338	0.366	0.388	0.358	0.388
Crops from annotation	**0.923**	**0.923**	**0.923**	**0.923**	**0.923**	**0.923**

Table 7: The results of the Triplet loss network when comparing each test sample with 10 embeddings from each class. Both raw accuracy and modified accuracy that ignores false positives from the Mask R-CNN are included.

	Accuracy			Modified accuracy		
Mask R-CNN crops	CS	L1 norm	L2 norm	CS	L1 norm	L2 norm
4608	**0.931**	**0.931**	**0.931**	**0.947**	**0.947**	**0.947**
1024	0.905	0.897	0.905	0.929	0.920	0.929
512	0.912	0.912	0.912	0.920	0.920	0.920
256	0.919	0.919	0.919	0.936	0.936	0.936
128	0.394	0.408	0.394	0.322	0.333	0.322
Crops from annotation	**0.957**	**0.957**	**0.957**	**0.957**	**0.957**	**0.957**

5 Discussion

"If humans have a hard time telling the difference, how much does it matter? But maybe a better question is: "What are we going to do with these detectors now that we have them?" A lot of the people doing this research are at Google and Facebook. I guess at least we know the technology is in good hands and definitely won't be used to harvest your personal information and sell it to.... wait, you're saying that's exactly what it will be used for?? Oh." - Joseph Redmon, Ali Farhadi

5.1 Data gathering

The images contain mostly fictional books, they are often colourful and of somewhat equal thickness (∼300 pages). Many images were photographed at the mathematics's library whose books were slightly different, particularly the thickness varied more. There were very thick ones and some so thin that even a human could have problems detecting them. They sometimes looked more like chopsticks if they had been pasted on to a background image. If we can not tell if it is a book or not, is it meaningful if we try learning them to a computer then? We know that different Computer Vision methods can outperform a person on specific tasks, so why not? The problem is if one actually puts chopsticks in the bookshelf. Then there is a good chance that they would be classified as books, giving us more false positives. If there are things like chopsticks in a bookshelf that is to be inventoried, it is probably a good idea to remove the chopsticks before the inventory if one wants to use this model.

For this thesis a rather small and uniform dataset has been used, although data augmentation has been performed, one should keep in mind that the model is built for classical bookshelf images (without chopsticks), containing up to 60-70 books, mostly taken perpendicular to the plane that the book spine form with small perspective alterations. If one uses another test dataset there is no guarantee that this model would have a mAP around 90%, which is clearly visible in Table 2 and 3 where two different datasets have been used and their results differ by approximately 10 percentage points. With more annotated images to train on, the difference might shrink. To get a more robust and general model, more samples of things that can look like books should be included in the dataset.

No study of the contribution of the data augmentation was made in this thesis, it would be a good idea to analyse. If a simpler training dataset gives the same result, it is unnecessary to do augmentation. Although, as the test images are rather similar to the training images, one should keep in mind that the data augmentation diversifies the data set, which might make the model better on general books, but possibly worse on these specific test datasets. Here, an attempt to make the model as general as it can get with a small dataset was made, and that is why the data augmentation was performed regardless of further analysis of its contribution. Though, if there would have been more time, this is one thing that should have been done.

5.2 Object Detection

A Mask R-CNN was used as it has some of the best results on COCO test-dev set. The other methods, YOLO, SSD and RetinaNet are faster than Mask R-CNN but typically loose some accuracy. The main objective for this thesis is to evaluate if it is possible to do such a book inventory, and to inventory objects in general. Thus, it is of utter importance that the objects are found and classified correctly, rather than doing it quickly.

Another important advantage of Mask R-CNN is the segmentation mask. Books are thin and might be skewed in a bookshelf. The other methods use bounding boxes that could have trouble segmenting each book. Two bounding boxes that have a large IoU are interpreted as the same book but with a mask it can tell that they are indeed two separate books. The Mask R-CNN trained for this thesis is still not good at this, but it could get better with more such samples in the training data.

It is hard to say exactly which image size the Mask R-CNN performs best on, but we can see that it is advantageous to use big images. It is not a big mAP difference between images with longest side of 4608 and 1024 for any of the test datasets, even though the longest side differs with a factor four and therefore the image size differs with a factor 16 for the 2D images. The difference between those and the 512 images is not big for the easier test dataset either. The reason that the difference in mAP between the 512 and 1024 images is bigger for the difficult test dataset than the easier might be because there in general are more books in each image in the difficult test dataset, which generally means that the books are represented by fewer pixels. We see that the mAP drops for smaller images, where books are represented by fewer pixels. Though this might not be the only reason for the difference in performance between the datasets. The books in the easier dataset with an image side of 512 are represented by fewer pixels than they are in the difficult dataset with image side 4608, but there is still a difference in mAP of nine percentage points. There are probably more factors than the book sizes that matters, for example the book shape. The very thin books can be troublesome to find even for humans, and there are many thin books in the difficult dataset.

From observations, this Mask R-CNN seems to work best for images where books are at least 400 pixels high and with a width that suggests that the book is more than 300 pages long. Books with various colours are probably advantageous as it is mainly what this model has been trained on. It is still not very good at separately segmenting many skewed books that are close together and that is probably because there are not a lot of such occurrences in the training data. Single skewed books are easier however. If one has books as described, with respect to distribution, shape and appearance, this model could surely be used to detect all the books in an image and be used for an automated inventory of books. Recognising each book, that is another cup of tea.

5.3 Recognition

The recognition of each book depends on what the Mask R-CNN crops. If the object detection method only thinks a part of a book is a book, the crop will not necessarily be valid while benchmarking as it requires an IoU of at least 0.5. These crops are automatically misclassified but it does not mean that the recognition method is bad. If these are ignored, the modified accuracy is given. In Table 5 one can find that the best result with categorical cross-entropy loss with Mask R-CNN crops is on images with the longest image side of 1024 pixels. The model with Triplet loss however, gets best result when using the biggest images as can be seen in Table 7 and 6. There is no obvious reason why they are good at different image sizes but it probably means that the two networks have found different important features that depend on the image size.

The network with triplet loss with mean embeddings gets slightly better results than the categorical cross-entropy loss CNN, and quite a lot better results when comparing with several samples. One might ponder whether the latter is a fair comparison since both CNN:s find weights that are good for a general sample. That is how they should be evaluated as well, not by comparing with several samples and take the most similar one. The classification with categorical cross-entropy loss is more similar to the classification with mean embeddings and to get a fair evaluation of which loss

function is best, one should compare with the mean embeddings. The classification based on 10 samples is included in this report to see how well the triplet loss method has succeeded in clustering the embeddings. If it was perfect, the mean and the 10 sample comparison would be more or less the same. In Table 7 and 6 we see that there is a difference, but it is not huge. In the appendix, two t-distributed Stochastic Neighbour Embedding (t-SNE) plots can be found, visualising how the classes have been clustered. t-SNE is a dimensionality reduction technique that converts the 512 dimensional embeddings into two dimensions which makes it possible for us to observe them in a modified space. As there are a lot of classes, it might be hard to read the labels of each point in the figures, but they give an idea of what the distribution looks like. There are also examples of a book whose embeddings have big variation and a few different books that have similar embeddings that can be found in the appendix.

Though the triplet loss network seems to work slightly better, it takes quite some time to train. It was trained roughly 40 times longer than the CNN with categorical cross-entropy loss, even though it had pretrained weights which the other did not have. Excessively time consuming training of models can be expensive, and that is a trade-off one has to do when choosing model, if the last few percentages in accuracy are more important than a quick method.

One interesting feature of the triplet loss results is that the network performs much better on crops from 256-pixel images than the categorical cross-entropy loss network does. This could be an indication that triplet loss can make the model more robust and scale invariant. It does not look like that for the images of size 128, but that can also be because they are too small for it to be possible to distinguish any important features at all.

5.4 Sources of Errors

Some of the images have books whose spine edges are not entirely distinguishable. Therefore, some of the front or back of the books might have been included in the annotations, or pieces of the book spines have been missed. This is considered as a relatively small source of error as there are only a few book spines that were hard to distinguish.

To evaluate the CNN:s, they were tested on the crops from the easy test dataset. This dataset did not have all the books that the networks had been trained on, so the evaluation of the CNN:s does not cover all the classes. We can not know how well they work on the remaining books, they might be very good but they can also be rather bad. To evaluate the CNN:s fairly, one should have test crops of all the books, which requires more data.

5.5 Future Work and Suggestions

To further improve the object detection, a suggestion is to use more data, both in quantity and diversity. More samples of negative objects could be included to make the model more robust.

To make a user friendly method, it should be able to use more or less in real time. The faster object detection models, YOLO, SSD and RetinaNet, that are described in Section 2.3.5 can be tried. Keep in mind that they only use bounding boxes which can be inconvenient for books. However, some people are have developed versions which they call Mask-YOLO, Mask-SSD and RetinaMask over the last few months, models that solves this problem.

There are many models to choose from and more or less the same amount of different backbones, loss functions and optimisers. In addition, hyper-parameters can be tuned in an infinite number of ways for every combination. With that said, there are always more things to try. There are

probably better models and parameters than those that have been used for this thesis but the main limitation is time, as the training processes are rather time-consuming. With other educated guesses, one could probably train even better models.

The recognition of books is based only on image features, but one can also try making a hybrid method as they did in [24] where they used a combination of image features and OCR. It increased the accuracy for them and the accuracy could possibly increase in this case as well.

Instead of comparing the distance between the triplet loss embeddings, one can use other alternatives. For instance one could try performing PCA on the embeddings and calculate some mean embedding for each class with a class variance in a few directions or experiment with support-vector machines. One can also try adding new books to the network as the network can create embeddings to any book image, regardless if the network is trained on the specific book or not. There is a possibility that it can take new books, which get unique embeddings, and then learn them as well. After one new book have been introduced and its embedding saved, another sample of the same book would in the best case scenario get a very similar embedding to the book it recently has learnt. The network in itself would be trained using supervised learning, but this kind of recognition could be unsupervised, as the new books would not need a specific label. The network would output which book or books the sample is most likely to belong to, hopefully one or a set of the new books that were added.

6 Conclusion

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

From a dataset of 50 annotated images of books, the state of the art object detection model Mask R-CNN was trained on more than 2000 augmentations of these images to detect book spines. It reached a mAP of 94% on one test dataset, but performs slightly worse on a harder dataset. It probably depends on the size of the image, or rather the pixel size of a book spine, among other factors. The object detection method cropped the book spines with the segmentation masks and the crops were classified with two different CNN:s with different loss functions and architectures. One used categorical cross-entropy loss and one used triplet loss, where the triplet loss network, based on the FaceNet, was slightly better in terms of accuracy but it takes a lot more time to train the network than it does for the simple CNN using categorical cross-entropy loss. The models designed in this Master Thesis could be used to perform an automated book inventory for simple cases. A simple case is for instance images that have been photographed more or less perpendicular to the plane that the book spines form, with colourful books that are fairly wide, standing upright or laying down. Each book should be sufficiently big in terms of pixels, i.e. at least 300-400 pixels long. That way the segmentation masks should be quite good which is essential for the recognition part. The recognition is fairly good, but can be improved. When ignoring invalid crops from the Mask R-CNN, they both reached an accuracy of 90%. The triplet loss and the categorical cross-entropy loss networks works similarly well, but triplet loss might be slightly better as it get good results for crops from smaller images too. Small images were harder to classify correctly in the categorical cross-entropy loss network.

7 References

[1] W. Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. https://github.com/matterport/Mask_RCNN, 2017.

[2] J. Brownlee. Deep learning for computer vision. https://machinelearningmastery.com/deep-learning-for-computer-vision/, 2019.

[3] D. M. Chen, S. S. Tsai, B. Girod, C.-H. Hsu, K.-H. Kim, and J. P. Singh. Building book inventories using smartphones. In ACM Multimedia, 2010.

[4] COCO, Common Objects in Context. Detection evaluation. http://cocodataset.org/#detection-eval, n.d.

[5] P. W. Code. Object detection on coco test-dev. https://paperswithcode.com/sota/object-detection-on-coco, 2019.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(1):142–158, Jan 2016.

[7] R. B. Girshick. Fast R-CNN. CoRR, abs/1504.08083, 2015.

[8] I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org.

[9] A. Gummeson. Prostate cancer classification using convolutional neural networks. Master's thesis, Centre for Mathematical Sciences, Lund Institute of Technology, Lund University, Lund, Sweden, 2016.

[10] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. CoRR, abs/1703.06870, 2017.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. CoRR, abs/1406.4729, 2014.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. CoRR, abs/1512.03385, 2015.

[13] R. Hermodsson and A. Wallin. Modelling intensive care unit length of stay and mortality using artificial neural networks. Master's thesis, Faculty of Engineering, LTH, Lund, Sweden, 2019.

[14] D. Kingma and J. Ba. Adam: A method for stochastic optimization. International Conference on Learning Representations, 12 2014.

[15] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. CoRR, abs/1708.02002, 2017.

[16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg. SSD: single shot multibox detector. CoRR, abs/1512.02325, 2015.

[17] Y. Liu, Y. Wang, S. Wang, T. Liang, Q. Zhao, Z. Tang, and H. Ling. Cbnet: A novel composite backbone network architecture for object detection. ArXiv, abs/1909.03625, 2019.

[18] A. Oppermann. Artificial intelligence vs. machine learning vs. deep learning. https://www.deeplearning-academy.com/p/ai-wiki-machine-learning-vs-deep-learning, 2019.

[19] O. Prakash. What are the key trade-offs between overfitting and underfitting? https://www.quora.com/What-are-the-key-trade-offs-between-overfitting-and-underfitting, 2018.

[20] N.-H. Quoc and W.-H. Choi. A framework for recognition books on bookshelves. volume 5754, pages 386–395, 09 2009.

[21] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. CoRR, abs/1506.02640, 2015.

[22] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. CoRR, abs/1804.02767, 2018.

[23] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. CoRR, abs/1506.01497, 2015.

[24] S. S. Tsai, D. M. Chen, H. Chen, C.-H. Hsu, K.-H. Kim, J. Pal Singh, and B. Girod. Combining image and text features: A hybrid approach to mobile book spine recognition. pages 1029–1032, 11 2011.

[25] D. Sandberg. Face recognition using tensorflow. https://github.com/davidsandberg/facenet, 2018.

[26] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. CoRR, abs/1503.03832, 2015.

[27] F. Shaikh. Deep learning vs. machine learning – the essential differences you need to know! https://www.analyticsvidhya.com/blog/2017/04/comparison-between-deep-learning-machine-learning/, 2017.

[28] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. CoRR, abs/1602.07261, 2016.

[29] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. International Journal of Computer Vision, 104(2):154–171, 2013.

[30] Unsplash. *Unsplash, Photos for everyone*. https://unsplash.com/, 2019.

[31] K. Wada. labelme: Image Polygonal Annotation with Python. https://github.com/wkentaro/labelme, 2016.

[32] X. Yang, D. He, W. Huang, A. Ororbia, Z. Zhou, D. Kifer, and C. L. Giles. Smart library: Identifying books on library shelves using supervised deep learning for scene text reading. In 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL), pages 1–4, June 2017.

8 Appendix

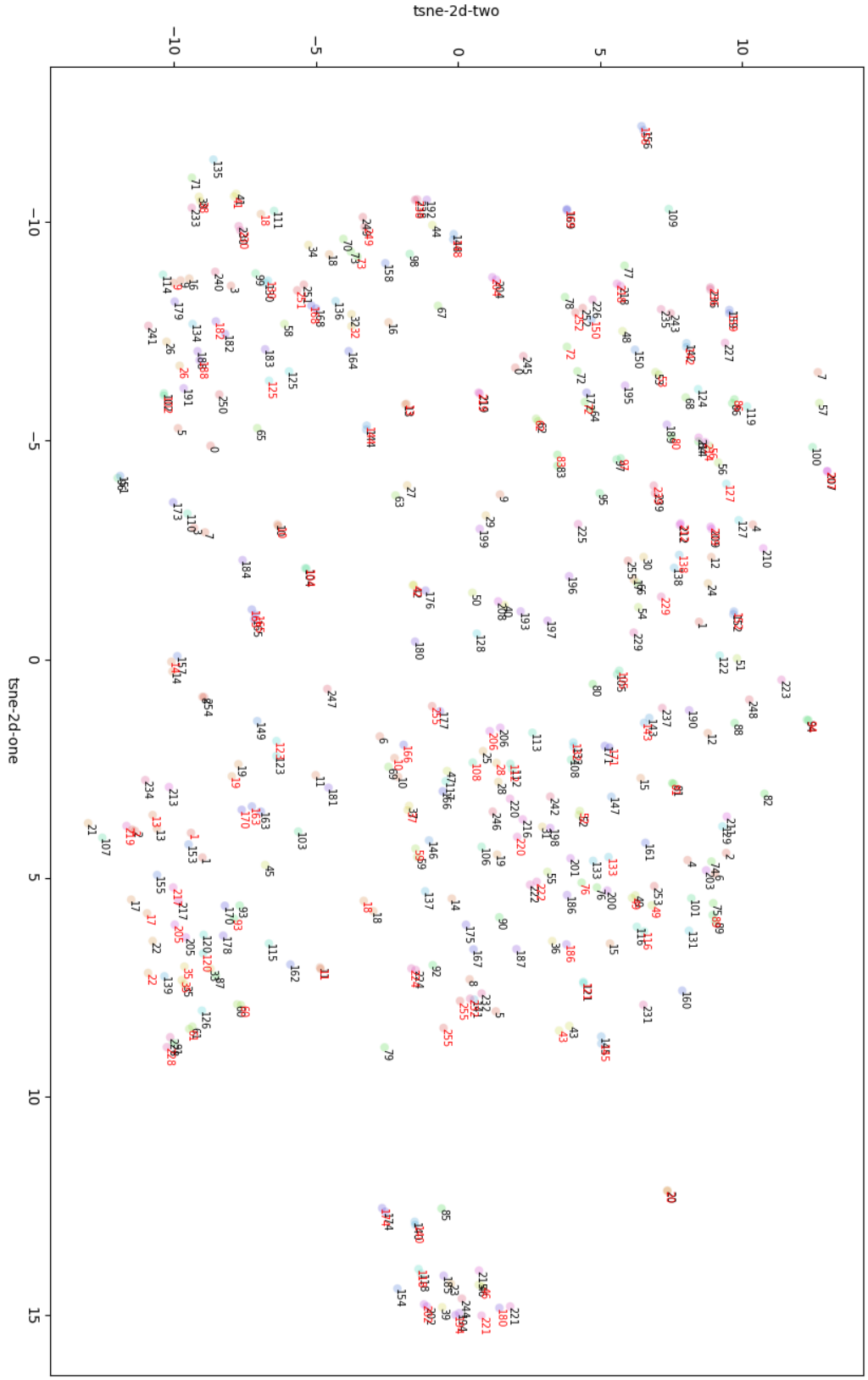

Figure 16: The t-SNE visualisation of the mean embeddings. The markers with black text correspond to training data and the ones with red text to test data.

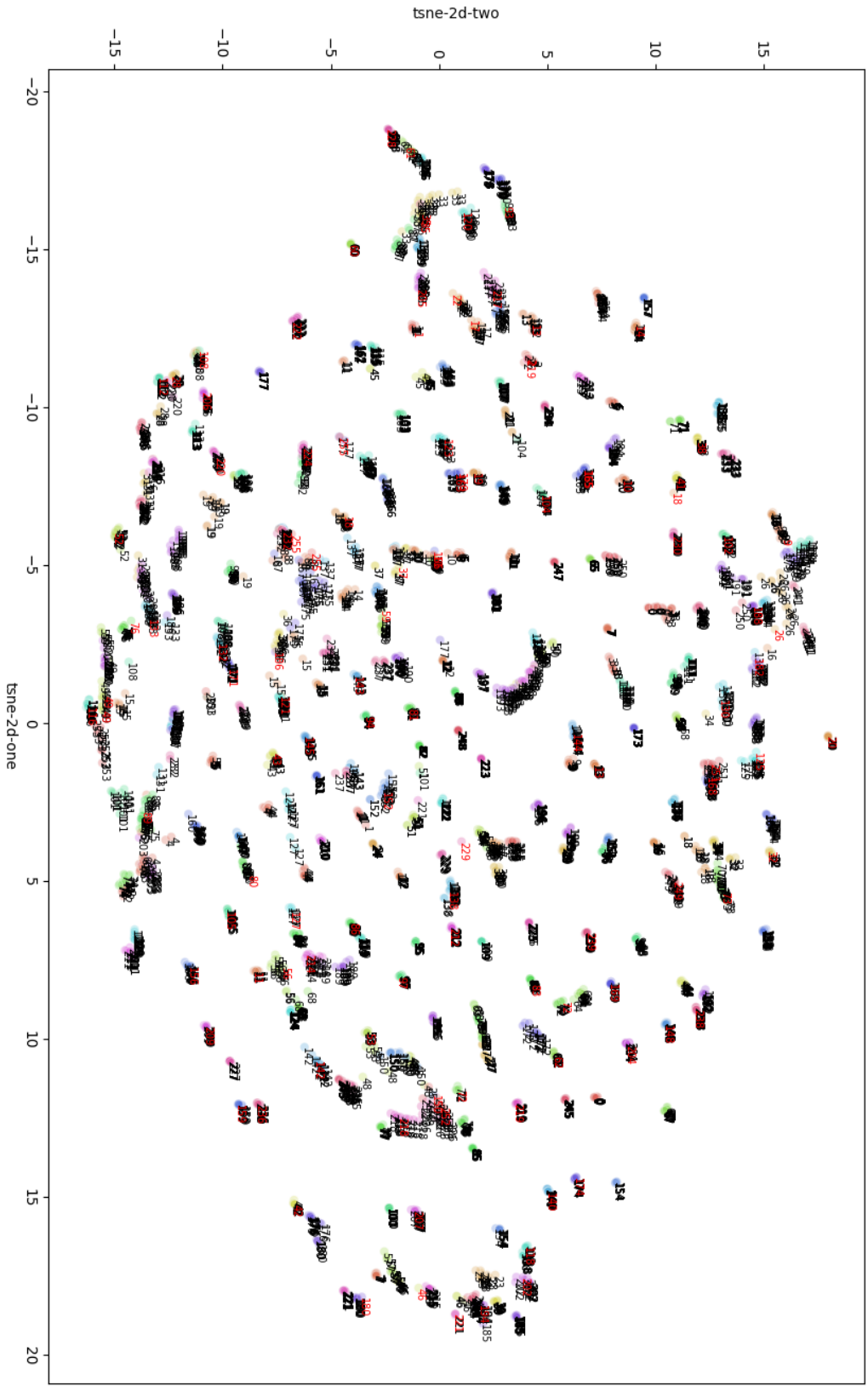

Figure 17: The t-SNE visualisation of several embeddings of each class. The markers with black text correspond to training data and the ones with red text to test data.

Figure 18: Some examples of a book with index 33 which has big variation in Figure 17, close to tsne-2d-one = -15, tsne-2d-two = 0.

Figure 19: Two different books that have similar embeddings. Their indices are 35 and 87 and they are also similar to the books in Figure 18. They can be found close to tsne-2d-one = -15, tsne-2d-two = 0 in Figure 17 and close to tsne-2d-one = 7, tsne-2d-two = -10 in Figure 16.

Figure 20: Two different books that are similar according to the triplet loss network with indices 175 and 167. They can be found close to tsne-2d-one = -4, tsne-2d-two = -6 in Figure 17, and close to tsne-2d-one = 7, tsne-2d-two = 0 in Figure 16.

Table 8: A list of all the books that were labeled with title and author.

Title	Author
Exit West	Mohsin Hamid
How to Get Filthy Rich in Rising Asia	Mohsin Hamid
The Reluctant Fundamentalist	Mohsin Hamid
The City Always Wins	Omar Robert Hamilton
Markings	Dag Hammarskjold
The Vegetarian	Han Kang
Miss Carters War	Sheila Hancock
Generation Loss	Elizabeth Hand
The Moravian Night	Peter Handke
A Sorrow Beyond Dreams	Peter Handke
We Are Pirates	Daniel Handler
A Case of Exploding Mangoes	Mohammed Hanif
Our Lady of Alice Bhatti	Mohammed Hanif
Uncommon Type	Tom Hanks
The A to Z of You and Me	James Hannah
Firefly Lane	Kristin Hannah
Fly Away	Kristin Hannah
The Mystery of Three Quarters	Sophie Hannah
The Art of Fielding	Chad Harbach
Painter of Silence	Georgina Harding
Enon	Paul Harding
Tinkers	Paul Harding
Far from the Madding Crowd 1	Thomas Hardy
Far from the Madding Crowd 2	Thomas Hardy
A Cat, a Hat and a Piece of String	Joanne Harris
Peaches for Monsieur le Cure	Joanne Harris
Dictator	Robert Harris
Lustrum	Robert Harris
Enchantments	Kathryn Harrison
Legends of the Fall	Jim Harrison
Envy	Kathryn Harrison
The Drafter	Kim Harrison
At Hawthorn Time	Melissa Harrison
Facial Justice	L. P. Hartley
The Go-Between	L. P. Hartley
Benediction	Kent Haruf
Our Souls at Night	Kent Haruf
All is Song	Samantha Harvey
The Asylum	John Harwood
Imagine Me Gone	Adam Haslett
Nowhere Ending Sky	Marlen Haushofer
Travesty	John Hawkes
The Scarlet Letter	Nathaniel Hawthorne
A student of weather	Elizabeth Hay

The Other Women: A Story About Three Transsexuals	Pamela Hayes
Love, Love Me Do	Mark Haysom
When Rain Clouds Gather	Bessie Head
Long Time, No See	Dermot Healy
The Saturday Evening Girls Club	Jane Healey
Blossoms and Shadows	Lian Hearn
Whats Eating Gilbert Grape	Peter Hedges
Standard Deviation	Katherine Heiny
Closing Time	Joseph Heller
Written in My Own Hearts Blood	Diana Gabaldon
The Mare	Mary Gaitskill
Veronica	Mary Gaitskill
The Kindly Ones	Jonathan Littell
Journey Through a Small Planet	Emanuel Litvinoff
The Paper Menagerie and Other Stories	Ken Liu
The Purple Swamp Hen and Other Stories	Penelope Lively
Thinks...	David Lodge
Novels and Stories	Jack London
White Fang	Jack London
China Blues	Ki Longfellow
Redemption in Indigo	Karen Lord
Lost Boi	Sassafras Lowrey
Under the Volcano	Malcolm Lowry
The Black Snow	Paul Lynch
The Secret Diary of Laura Palmer	Jennifer Lynch
Ports of Call	Amin Maalouf
In a Land of Paper Gods	Rebecca Mackenzie
And This Is True	Emily Mackie
Midwinter Break	Bernard MacLaverty
Island	Alistair MacLeod
All the Beloved Ghosts	Alison MacLeod
If Not for You	Debbie Macomber
The Lady from Tel Aviv	Rabai al-Madhoun
The Undertaking	Audrey Magee
Palace Walk	Naguib Mahfouz
Mind of an Outlaw: Selected Essays	Norman Mailer
The Naked and the Dead	Norman Mailer
Company of Liars	Karen Maitland
The Vanishing Witch	Karen Maitland
The Abundance: A Novel	Amit Majmudar
You Deserve Nothing	Alexander Maksik
Kintu	Jennifer Nansubuga Makumbi
The Japanese Lover	Rani Manicka
The Rice Mother	Rani Manicka
Buddenbrooks	Thomas Mann
Three Amazing Things About You	Jill Mansell
The Collected Stories of Katherine Mansfield	Katherine Mansfield

Wolf Hall	Hilary Mantel
The Assassination of Margaret Thatcher	Hilary Mantel
Like a Mule Bringing Ice Cream to the Sun	Sarah Ladipo Manyika
The Infatuations	Javier Marias
Thus Bad Begins	Javier Marias
You Dont Have to Live Like This: A Novel	Benjamin Markovits
Matterhorn	Karl Marlantes
Lost Girls	Angela Marsons
The High Mountains of Portugal	Yann Martel
Life of Pi	Yann Martel
Self	Yann Martel
Nanjing Requiem	Ha Jin
A Hero Born	Jin Yong
Fortune Smiles	Adam Johnson
Train Dreams	Denis Johnson
A Lady Cyclists Guide to Kashgar	Suzanne Joinson
The Hundred-Year-Old Man	Jonas Jonasson
Journey to the West	Wu Chengen
An Enormously English Monsoon Wedding	Christina Jones
Never Greener	Ruth Jones
The Uninvited Guests	Sadie Jones
Fear of Dying	Erica Jong
Fear of Flying	Erica Jong
The Living	Anjali Joseph
The Limits of Enchantment	Graham Joyce
Finnegans Wake	James Joyce
A Portrait of the Artist as a Young Man	James Joyce
The Restored Finnegans Wake	James Joyce
Perfect	Rachel Joyce
The Love Song of Miss Queenie Hennessy	Rachel Joyce
The Music Shop	Rachel Joyce
The First Bad Man	Miranda July
No One Belongs Here More Than You	Miranda July
The Unlikely Pilgrimage of Harold Fry	Rachel Joyce
The Traitors Niche	Ismail Kadare
The Trial	Franz Kafka
The Report	Jessica F Kane
The Private Life of Mrs. Sharma	Ratika Kapur
The Other	Ryszard Kapuscinski
The Invoice	Jonas Karlsson
Second Person Singular	Sayed Kashua
Native	Sayed Kashua
Born Weird	Andrew Kaufman
Come to the Edge	Joanna Kavenna
The House of the Sleeping Beauties	Yasunari Kawabata
Manazuru	Hiromi Kawakami
The Master of Go	Yasunari Kawabata

The Long Room	Francesca Kay
Mountains of the Moon	I. J. Kay
Measuring the World	Daniel Kehlmann
Lake Wobegon summer 1956	Garrison Keillor
Leaving Home	Garrison Keillor
Pigeon English	Stephen Kelman
The Daughters of Mars	Thomas Keneally
Schindlers List	Thomas Keneally
The Moment	Douglas Kennedy
Serious Sweet	A. L. Kennedy
The Woman in the Fifth	Douglas Kennedy
Suddenly, a Knock on the Door	Etgar Keret
Knellers Happy Campers	Etgar Keret
The Dharma Bums	Jack Kerouac
Doctor Sax	Jack Kerouac
Orpheus Emerged	Jack Kerouac
Road Novels 1957-1960	Jack Kerouac
Fatelessness	Imre Kertesz
Kaddish for a Child Not Born	Imre Kertesz
One Flew Over the Cuckoos Nest	Ken Kesey
A Good Country	Laleh Khadivi
The sirens of Baghdad	Yasmina Khadra
In Praise of Hatred	Khaled Khalifa
No Knives in the Kitchens of This City	Khaled Khalifa
Thinner Than Skin	Uzma Aslam Khan
Everything I Dont Remember	Jonas Hassen Khemiri
The Broken Mirrors: Sinalcol	Elias Khoury
My Name is Adam	Elias Khoury
The Invention of Wings	Sue Monk Kidd
Snapper	Brian Kimberling
Flight Behavior	Barbara Kingsolver
Shopaholic Takes Manhattan	Sophie Kinsella
The Wake	Paul Kingsnorth
Small Wonder	Barbara Kingsolver
Villa America	Liza Klaussmann
I Dont Care About Your Band	Julie Klausner
Orphan Train	Christina Baker Kline
Boyhood Island	Karl Ove Knausgaard
Dancing in the dark	Karl Ove Knausgaard
My Struggle Book 1	Karl Ove Knausgaard
My Struggle Book 2	Karl Ove Knausgaard
Some Rain Must Fall	Karl Ove Knausgaard
Ill Be Right There	Shin Kyung-sook
Super Sad True Love Story	Gary Shteyngart
The Pilots Wife	Anita Shreve
Stella Bain	Anita Shreve
The Stars Are Fire	Anita Shreve

Big Brother	Lionel Shriver
The Mandibles	Lionel Shriver
Property	Lionel Shriver
We Need to Talk About Kevin	Lionel Shriver
Quo Vadis	Henryk Sienkiewicz
Ice Candy Man	Bapsi Sidhwa
The Crow Eaters	Bapsi Sidhwa
More Happy Than Not	Adam Silvera
The Abominable	Dan Simmons
The Bronze Horseman	Paullina Simons
In-Flight Entertainment	Helen Simpson
The Rosie Effect	Graeme Simsion
Animals People	Indra Sinha
Noahs Compass	Anne Tyler
The Beginners Goodbye	Anne Tyler
Reunion	Fred Uhlman
Daniel Stein, Interpreter	Ludmila Ulitskaya
Licks of Love	John Updike
Villages	John Updike
Rabbit Is Rich	John Updike
Rabbit at Rest	John Updike
Rabbit Redux	John Updike
Rabbit, Run	John Updike
The Witches of Eastwick	John Updike
The Merman	Carl-Johan Vallgren
Acceptance	Jeff VanderMeer
Caribou Island	David Vann
The Way Of The Women	Marlene van Niekerk
Goat Mountain	David Vann
Legend of a Suicide	David Vann
Annihilation	Jeff VanderMeer
Authority	Jeff VanderMeer
The Bad Girl	Mario Vargas Llosa
The Discreet Hero	Mario Vargas Llosa
The Feast of the Goat	Mario Vargas Llosa
The War of the End of the World	Mario Vargas Llosa
The Magic of Saida	M. G. Vassanji
Cutting for Stone	Abraham Verghese
The River of No Return	Bee Ridgway
The Pearl Sister	Lucinda Riley
The Other Typist	Suzanne Rindell
Shantaram	Gregory David Roberts
If Ever I Fall	S. D. Robertson
Housekeeping	Marilynne Robinson
Gilead	Marilynne Robinson
Lila	Marilynne Robinson
When I Was a Child I Read Books	Marilynne Robinson

The House at the Edge of the World	Julia Rochester
Red April	Santiago Roncagliolo
Jonathan Unleashed	Meg Rosoff
The Dark Flood Rises	Margaret Drabble
The Count of Monte Cristo	Alexandre Dumas
In the company of the courtesan	Sarah Dunant
Dunbar	Edward St Aubyn
The Lie	Helen Dunmore
The Adulterants	Joe Dunthorne
Finding Fraser	kc dyer
Tarantula	Bob Dylan
Regeringsrattens arsbok 1943	Domstolsverket
Regeringsrattens arsbok 1944	Domstolsverket
Regeringsrattens arsbok 1945	Domstolsverket
Regeringsrattens arsbok 1946	Domstolsverket
Regeringsrattens arsbok 1947	Domstolsverket
Regeringsrattens arsbok 1948	Domstolsverket
Regeringsrattens arsbok 1949	Domstolsverket
Regeringsrattens arsbok 1950	Domstolsverket
Regeringsrattens arsbok 1951	Domstolsverket
Regeringsrattens arsbok 1952	Domstolsverket
Regeringsrattens arsbok 1953	Domstolsverket
Regeringsrattens arsbok 1954	Domstolsverket
Regeringsrattens arsbok 1955	Domstolsverket
Regeringsrattens arsbok 1956	Domstolsverket
Regeringsrattens arsbok 1957	Domstolsverket
Stationary Stochastic Processes	Lindgren, Rootzen, and Sandsten
Introduction to the Finite Element Method	Ottosen & Petersson
Flerdimensionell analys	Jonas Mansson, Patrik Nordbeck
Flerdimensionell analys ovningar	Jonas Mansson, Patrik Nordbeck
Kvantvarldens fenomen	Gunnar Ohlen
Linjar algebra	Gunnar Sparr
Endimensionell analys	Jonas Mansson, Patrik Nordbeck
Artificial Intelligence: A Modern Approach	Stuart J. Russell, Peter Norvig
Sannolikhetsteori och statistikteori med tillampningar	Gunnar Blom
Arvet Efter Dig	Jojo Moyes
Toner i Natten	Jojo Moyes
Sista Brevet till din Alskade	Jojo Moyes
Foretagsekonomi 100	Per-Hugo Skarvad, Jan Olsson
Industriell ekonomi	Mona Becker
ovningar i industriell ekonomi	Jan Moller
Hallbar Utveckling	Grondahl, Svanstrom
The Lost Boy	Dave Pelzer
Atlantis	David Gibbins

Master's Theses in Mathematical Sciences 2019:E62
ISSN 1404-6342

LUTFMA-3394-2019

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/